

Simulation Foundations, Methods and Applications

Tuncer Ören
Bernard P. Zeigler
Andreas Tolk *Editors*

Body of Knowledge for Modeling and Simulation

A Handbook by the Society
for Modeling and Simulation
International



THE SOCIETY FOR
MODELING & SIMULATION
INTERNATIONAL



Springer

Simulation Foundations, Methods and Applications

Series Editor

Andreas Tolk, The MITRE Corporation, Charlottesville, VA, USA

Advisory Editors

Roy E. Crosbie, California State University, Chico, CA, USA

Tony Jakeman, Australian National University, Canberra, ACT, Australia

Axel Lehmann, Universität der Bundeswehr München, Neubiberg, Germany

Stewart Robinson, Loughborough University, Loughborough, Leicestershire, UK

Bernard P. Zeigler, University of Arizona, Tucson, AZ, USA

The modelling and simulation community extends over a range of diverse disciplines and this landscape continues to expand at an impressive rate. Modelling and simulation is fundamentally a computational tool which has an established record of significantly enhancing the understanding of dynamic system behaviour on one hand, and the system design process on the other. Its relevance is unconstrained by discipline boundaries. Furthermore, the ever-increasing availability of computational power makes feasible applications that were previously beyond consideration.

Simulation Foundations, Methods and Applications hosts high-quality contributions that address the various facets of the modelling and simulation enterprise. These range from fundamental concepts that are strengthening the foundation of the discipline to the exploration of advances and emerging developments in the expanding landscape of application areas. The underlying intent is to facilitate and promote the sharing of creative ideas across discipline boundaries. The readership will include senior undergraduate and graduate students, modelling and simulation professionals and research workers.

Inasmuch as a model development phase is a prerequisite for any simulation study, there is an expectation that modelling issues will be appropriately addressed in each presentation. Incorporation of case studies and simulation results will be strongly encouraged.

Titles can span a variety of product types, including but not exclusively, textbooks, expository monographs, contributed volumes, research monographs, professional texts, guidebooks and other references.

These books will appeal, varyingly, to senior undergraduate and graduate students, and researchers in any of a host of disciplines where modelling and simulation has become (or is becoming) a basic problem-solving tool. Some titles will also directly appeal to modelling and simulation professionals and practitioners.

Tuncer Ören • Bernard P. Zeigler •
Andreas Tolk
Editors

Body of Knowledge for Modeling and Simulation

A Handbook by the Society
for Modeling and Simulation
International

Editors

Tuncer Ören
School of Electrical Engineering
and Computer Science
University of Ottawa
Ottawa, ON, Canada

Bernard P. Zeigler
Department of Electrical and Computer
Engineering
University of Arizona
Tucson, AZ, USA

Andreas Tolk
The MITRE Corporation
Charlottesville, VA, USA

ISSN 2195-2817

ISSN 2195-2825 (electronic)

Simulation Foundations, Methods and Applications

ISBN 978-3-031-11084-9

ISBN 978-3-031-11085-6 (eBook)

<https://doi.org/10.1007/978-3-031-11085-6>

© The Editor(s) (if applicable) and The Author(s), under exclusive license to Springer Nature Switzerland AG 2023

This work is subject to copyright. All rights are solely and exclusively licensed by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors, and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

*To the many distinguished simulationist
colleagues
whom I had the chance to collaborate with
since 1963
and to Füsün, my wife and my lifelong friend.*

Tuncer Ören

*To all our fellow participants in the modeling
and simulation adventure:
may we help the world keep up more
competently with the technology it keeps
advancing.*

Bernard P. Zeigler

*To the many pioneers of modeling and
simulation who paved the way to
make us a scientific discipline, and to my
family—Andrea, Florian, and
Christopher—who had to share me so often
with my passion for these scientific endeavors.*

Andreas Tolk

A handwritten signature in black ink, appearing to be 'B. Zeigler', written in a cursive style.

Preface

Any scientific Body of Knowledge is a comprehensive and concise representation of concepts, terms, and activities needed to explain a professional domain by representing the common understanding of relevant professionals and professional associations. Defining this body for Modeling and Simulation (M&S) is essential for the discipline of M&S. These concepts must include the science-philosophical foundations and implications, the understanding what models are and how their implementation in form of simulations can be used to support the many application domains, such as gaining experience for training and education, as well as experimentation for analysis, design, control, and optimization, within increasing number of computational disciplines.

But is Modeling and Simulation really a discipline? To some people, *simulation* is a very useful technique for representing a system under study to enable computational experimentation with a view to improve system performance. In this perspective, *modeling*, the representation itself, is of secondary importance—merely a necessary means to an end. For other users, simulation provides a virtual environment that allows to train people. The defense simulation community is a good example for this type of simulation use, but also the aviation community using flight simulators to educate and qualify their pilots. These users look at simulation as a powerful computational tool.

In contrast, underlying the SCS Modeling and Simulation Body of Knowledge (M&SBoK) is the assertion that there is a discipline called *Modeling and Simulation* (M&S). Moreover, this discipline provides visibility into the holistic nature, and the conjoint activities, of model creation and simulation experimentation. At the core of the M&S discipline is the identification of the elements manipulated by its associated activities: real system data, experimental frame, model, and simulator, as well as the relationships that must bind these components together to form a meaningful composition.

There is a huge paradigm shift from M&S as a computational tool to the M&S as discipline world view. Taking this shift, the M&S framework ontology (the four elements and their relations) effectively lays the foundation for computational experiments, clearly stating boundaries, and interactions, of systems, data, and representation. This shifts the focus from simulation to modeling, placing the model at the center, making the model the curated artifact of knowledge that must be

maintained, enhanced, and reused over time. This viewpoint allows the application of simulation in many computational sciences to help gaining new knowledge by creating numerical insight into the dynamic behavior of the modeled entities, the use of M&S as an epistemological tool.

These views are not mutually exclusive. Contrarily, they represent multiple facets of the variety how M&S is and can be used. M&S supports many disciplines, computationally as well as epistemologically. The simulation engineer must be aware of the whole picture to serve their communities best. They must understand the application domains and must be able to support the best conceptualization and capture this in a model that is implemented as a simulation. If the application domain uses IT support, the simulation engineer should be aware of interfaces that can support the data information exchange needs. The M&SBoK provides a first collection of such knowledge and surely needs to be a living document that is augmented over time.

In this initial Guide to M&SBoK, Chap. 1 sets out the concepts of the M&S framework ontology that lay the groundwork for subsequent discussion. Chapter 2 covers the core areas of M&S and provides an overall big picture portrait of this emerging discipline and how it supports other knowledge domains. Chapter 3 covers the traditional view of simulation as experimentation. Indeed, there is no other discipline that can provide powerful simulations providing numerical insight into complex dynamic systems. And yet, the epistemology of M&S is the brain power that enables these tools. Simulation is the muscle; modeling is the soul!

Chapters 4 and 5 introduce simulation as *experience*, both in the technical and entertainment arenas. Chapters 6–10 cover the internals of the M&S disciple, its mechanics, ethics, and economics, while Chaps. 11–16 concern the external relationships, how M&S is taking its place among, and increasingly central to, the recognized disciplines in science, technology, and the arts. Finally, Chaps. 17–19 review the development of M&S over time and set forth the trends, aspirations, and challenges of the future.

It is expected that the M&SBoK will grow over time. This first version is a first set of core concepts, but as the application domains of simulation grow, so will the body of knowledge. We already see growing fields that need to be addressed in more detail, such as complexity, deep uncertainty, and quantum simulation. We assume that in the next iteration, we will see how M&S can be increasingly used to address complex system, which comprises a variety of heterogenous entities, often highly interdependent and connect in nonlinear fashion. We assume to see an increasing use to address the challenge of deep uncertainty in operations research, which requires multi-model approaches and a new paradigm to conduct simulation-based optimization to understand the topology of the solution space instead of looking for point-solutions. And finally, with quantum computing becoming increasingly available, simulation engineers have to address how we use this new resource: like we developed concepts for parallel and distributed simulation we will have to address quantum computing-based simulation soon to be ready as the community of simulation engineers when the time comes.

We like to express our thanks to the Virginia Modeling, Analysis, and Simulation Center (VMASC) of Old Dominion University, Norfolk, Virginia, as they committed to provide a website hosting any errata as well as supplemental material for the M&SBoK. The website can be accessed at <http://vmasc.org/partnerships/msbok>.

In summary, we are convinced that we are indeed an M&S discipline. We have our professional societies, our journals, our code of ethics, and our common knowledge. We address many of these concepts, we are grateful to the many contributors to create this foundational initial version, and we are excited about the future and the new insights to come and to be integrated to witness to the growing contributions of M&S to solve the challenges of our society.

Ottawa, ON, Canada
Tucson, AZ, USA
Charlottesville, VA, USA
March 2022

Tuncer Ören
Bernard P. Zeigler
Andreas Tolk

Contents

1	Preliminary	1
	Tuncer Ören, Bernard P. Zeigler, and Thorsten Pawletta	
2	M&S Bok Core Areas and the Big Picture	21
	Tuncer Ören, Umang Kant, Mayank Sing, Paul Fishwick, Mamadou Kaba Traoré, Lin Zhang, Yuanjun Laili, Bernard P. Zeigler, Andreas Tolk, Gregory Zacharewicz, Mohammad S. Obaidat, and Balqies Sadoun	
3	Simulation as Experimentation	77
	Tuncer Ören, Paul K. Davis, Rhys Goldstein, Azam Khan, Laurent Capocchi, Maâmar El-Amine Hamri, Navonil Mustafee, Alison L. Harper, Baocun Hou, Bo Hu Li, and Yang Liu	
4	Simulation as Experience to Enhance Three Types of Skills	121
	Tuncer Ören, Umut Durak, Ernest H. Page, Andreas Tolk, and Saikou Y. Diallo	
5	Simulation Games	141
	Rodrigo Pereira dos Santos and Esteban Walter Gonzalez Clua	
6	Infrastructure	149
	Margaret L. Loper, Tuncer Ören, Cláudio Gomes, Valdemar Vicente Graciano Neto, and Ernest H. Page	
7	Reliability and Quality Assurance of M&S	167
	Tuncer Ören, Valdemar Vicente Graciano Neto, Paul K. Davis, and Bernard P. Zeigler	
8	Ethics	205
	Nico Formanek, Juan Manuel Durán, Paul K. Davis, Andreas Tolk, and Tuncer Ören	
9	Enterprise Modeling and Simulation	221
	Hezam Haidar, Nicolas Daclin, Gregory Zacharewicz, and Guy Doumeingts	

10	Maturity and Accreditation	249
	Tuncer Ören and Margaret L. Loper	
11	Supporting Computer Domains	255
	Jean François Santucci, Laurent Capocchi, Tuncer Ören, Saurabh Mittal, Bo Hu Li, Lin Zhang, Ting Yu Lin, Yuanjun Laili, and Claudia Szabo	
12	Synergies of Soft Computing and M&S	287
	Jean François Santucci, Laurent Capocchi, Tuncer Ören, Claudia Szabo, and Valdemar Vicente Graciano Neto	
13	Supporting Science Areas	311
	Bernard P. Zeigler, Paul Wach, Laurent Capocchi, Paul Weirich, Mohammad S. Obaidat, Balqies Sadoun, and Claudia Szabo	
14	Supporting Engineering Areas	353
	Andrea D’Ambrogio, Chen Yang, and Hessam S. Sarjoughian	
15	Supporting Social Science and Management Areas	373
	Paul K. Davis	
16	Philosophy and Modeling and Simulation	383
	Andreas Tolk, Ernest H. Page, Valdemar Vicente Graciano Neto, Paul Weirich, Nico Formanek, Juan Manuel Durán, Jean François Santucci, and Saurabh Mittal	
17	History of Simulation	413
	Bernard P. Zeigler, Breno Bernard Nicolau de França, Valdemar Vicente Graciano Neto, Raymond R. Hill, Lance E. Champagne, and Tuncer Ören	
18	Core Research Areas	435
	Paul Fishwick, Saikou Y. Diallo, Umut Durak, Baocun Hou, Bo Hu Li, Chunhui Su, Yanguang Wang, Lin Zhang, Xu Xie, Longfei Zhou, Bernard P. Zeigler, Thorsten Pawletta, Hendrik Folkerts, and Saurabh Mittal	
19	Trends, Desirable Features, and Challenges	471
	Bo Hu Li, Wenhui Fan, and Lin Zhang	
	Appendix A: Terminology and Other Reference Documents	495
	Appendix B: Bios of the Contributors	515
	Index	543

Contributors

- Laurent Capocchi** University of Corsica, Haute-Corse, Corte, France
- Lance E. Champagne** Air Force Institute of Technology, Dayton, OH, USA
- Esteban Walter Gonzalez Clua** Universidade Federal Fluminense, Niterói, Brazil
- Andrea D'Ambrogio** University of Roma Tor Vergata, Rome, Italy
- Nicolas Daclin** IMT-Mines Ales, Alès, France
- Paul K. Davis** The RAND Corporation and the Pardee RAND Graduate School, Santa Monica, CA, USA
- Saikou Y. Diallo** Old Dominion University, Norfolk, VA, USA
- Guy Doumeingts** Bordeaux University, Nouvelle-Aquitaine, France
- Umut Durak** German Aerospace Center, Cologne, Germany
- Juan Manuel Durán** Delft University of Technology, Delft, Netherlands
- Wenhui Fan** Tsinghua University, Beijing, China
- Paul Fishwick** University of Texas at Dallas, Richardson, TX, USA
- Hendrik Folkerts** University of Applied Sciences in Wismar, Wismar, Germany
- Nico Formanek** High Performance Computing Center, Stuttgart, Germany
- Rhys Goldstein** Autodesk Research, Toronto, ON, Canada
- Cláudio Gomes** Aarhus University, Aarhus, Denmark
- Valdemar Vicente Graciano Neto** Federal University of Goiás, Goiania, Brazil
- Hezam Haidar** INTEROP-VLap, Brussels, Belgium
- Maâmar El-Amine Hamri** Aix-Marseille University (LSIS), Marseille, France
- Alison L. Harper** University of Exeter Medical School, Exeter, UK
- Raymond R. Hill** Air Force Institute of Technology, Dayton, OH, USA

-
- Baocun Hou** Midea Cloud Tech Co., Ltd, Jilin, Beijing, China
- Umang Kant** Delhi Technological University, New Delhi, India
- Azam Khan** Trax.Co, Toronto, ON, Canada
- Yuanjun Laili** Beihang University, Beijing, China
- Bo Hu Li** Beihang University, Beijing, China
- Ting Yu Lin** Beijing Simulation Center, Beijing, China
- Yang Liu** CASICloud-Tech Co., Ltd., Haidian, China
- Margaret L. Loper** Georgia Tech Research Institute, Atlanta, GA, USA
- Saurabh Mittal** The MITRE Corporation, Dayton, OH, USA
- Navonil Mustafee** University of Exeter Business School, Exeter, UK
- Breno Bernard Nicolau de França** Universidade Estadual de Campinas, Campinas, Brazil
- Mohammad S. Obaidat** University of Texas-Permian Basin, Odessa, TX, USA
King Abdullah II School of Information Technology, University of Jordan, Jordan
and University of Science and Technology, Amman, Jordan
- Tuncer Ören** University of Ottawa, Ottawa, ON, Canada
- Ernest H. Page** The MITRE Corporation, McLean, VA, USA
- Thorsten Pawletta** University of Applied Sciences in Wismar, Wismar, Germany
- Balqies Sadoun** College of Engineering, Al Balqa' Applied University, Al Salt, Jordan
- Rodrigo Pereira dos Santos** Federal University of the State of Rio de Janeiro, Rio de Janeiro, Brazil
- Jean François Santucci** University of Corsica, Corte, France
- Hessam S. Sarjoughian** Arizona State University, Tempe, AZ, USA
- Mayank Sing** Consilio Research Lab, Noida, India
- Chunhui Su** CASICloud-Tech Co., Ltd., Beijing, China
- Claudia Szabo** University of Adelaide, Adelaide, SA, Australia
- Andreas Tolk** The MITRE Corporation, Charlottesville, VA, USA
- Mamadou Kaba Traoré** University of Bordeaux, Bordeaux, France
- Paul Wach** Virginia Tech, Blacksburg, VA, USA
- Yanguang Wang** CASICloud-Tech Co., Ltd., Beijing, China

Paul Weirich University of Missouri, Columbia, MO, USA

Xu Xie National University of Defense Technology, Changsha, China

Chen Yang Beijing Institute of Technology, Beijing, China

Gregory Zacharewicz IMT-Mines Alès, Alès, France

Bernard P. Zeigler University of Arizona, Tucson, AZ, USA

Lin Zhang Beihang University, Beijing, China

Longfei Zhou Duke University Medical Center, Durham, NC, USA



Preliminary

1

Tuncer Ören , Bernard P. Zeigler , and Thorsten Pawletta 

Abstract

In this chapter, we provide an introductory view for the scope of the SCS M&S Body of Knowledge, including the terminology. We provide a rationale for the theoretical basis of M&S and give an overview of the modeling and simulation framework (MSF) applied in many contributions, followed by the basic system entity structure (SES) concepts.

Keywords

Modeling and simulation · Discrete event systems specification (DEVS) · Modeling and simulation framework (MSF) · System entity structure (SES)

1.1 Scope

Tuncer Ören

The term simulation, based on the concept of similarity, has been used in English since mid-fourteenth century. Hence, simulation has non-technical as well as technical meanings. Accordingly, it has many definitions. A collection of about 100

T. Ören (✉)
University of Ottawa, Ottawa, ON, Canada
e-mail: toren@uottawa.ca

B. P. Zeigler
University of Arizona, Tucson, USA
e-mail: zeigler@rtsync.com

T. Pawletta
University of Applied Sciences in Wismar, Wismar, Germany
e-mail: thorsten.pawletta@hs-wismar.de

definitions of the term simulation is compiled and categorized in three groups and nine subgroups by Ören [1]. In a sequel article, a critical review of these definitions was given [2].

In a Body of Knowledge document, it is imperative to delimit the scope of the main concept. Therefore, the meaning(s) of the term “simulation” as used in this document is clarified based on Ören [2].

From a pragmatic point of view, based on the purpose of its use, simulation has three aspects:

- (1) Perform *experiments*,
- (2) Gain *experience*
for training to gain/enhance any one of the three types of skills, or for entertainment, and
- (3) Imitation, pretense.

Only the experiment and experimentation aspects are within the scope of this study.

From experimentation aspect: “*Simulation is performing goal-directed experiments with models of dynamic systems.*”

Purpose of simulation, as well as definitions and explanations of different types of experiments—outlined in Ören [2]—is elaborated later.

From the experience perspective, simulation has two distinct usages for *training* and for *entertainment*:

“Simulation is providing experience under controlled conditions for training, i.e., for gaining/enhancing competence in one of the three types of skills:

- (1) motor skills (virtual simulation or use of simulators),
- (2) decision and/or communication skills (constructive simulation such as business games, war games, or peace games; aka serious games), and
- (3) operational skills (live simulation).”

Experience through gaming simulations can be used for training as well as for entertainment purposes.

Simulation has many aspects. A recent publication lists 750 types of simulation [3].

1.2 Terminology

Bernard P. Zeigler

After several decades of development, there are still a wide variety of modeling and simulation terms and associated concepts with multiple interpretations. This variety derives from different historical streams of development of Modeling and Simulation (M&S) within different contexts (whether industrial, governmental, or military) or disciplinary (whether in “hard” or “soft” science or engineering). The premise behind the need for a body of knowledge for M&S is that there is some

core set of concepts that identify its activities as different from others and that are common no matter in which context they are employed. The reference list [4] contains 29 dictionaries of modeling and simulation. Some definitions can provide a good “first approximation” to understanding a term and how it is used, and they lack the precision and rigor that a BoK should aspire to. Therefore, this BoK employs a Framework of M&S (reviewed in Sect. 1.4) to map out the basic entities and their relationships employed in M&S and at the same time, providing a theory-based set of definitions employed in the framework. Terms such as “model” and “simulator” are often loosely used in current practice but have a very sharp meanings in the framework. Therefore, it is important to understand what is included and excluded by the definitions. (This is especially true if you have some experience in M&S and are likely to associate (or prefer) meanings that are different from those developed here.) Based on the Modeling and Simulation Framework (MSF), the basic issues and problems encountered in performing M&S activities and using their vocabulary can be better understood and coherent solutions developed. Understanding the MSF core concepts and employing the associated terminology will help everyone involved in a simulation modeling project such as analysts, programmers, managers, and users to better carry out their tasks and communicate with each other.

1.3 Rationale for Theoretical Basis of M&S

Bernard P. Zeigler

“An established body of knowledge (BoK) is one of the pillars of an established discipline” [5]. The BoK establishes a kernel of topics that categorically characterize the discipline. When sufficiently mature, a comprehensive theory of the domain provides an essential framework to define a kernel of topics and to organize these topics in a meaningful way. Further, the framework and its underlying theory provide a sound foundation for conduct of activities in the discipline.

At this point in time, it has been asserted that “Theory of Modeling and Simulation (1976) gives a theory for simulation that is based on general system theory and this theory is considered the only major theory for simulation. This book showed that simulation has a solid foundation and is not just some ad hoc way of solving problems” [6]. Furthermore, “Theory of Modeling and Simulation (1976) is a major reference for modeling formalisms, particularly the Discrete Event System Specification (DEVS). ... We mention the System Entity Structures and Model Base (SES/MB) framework as breakthrough in this field [Model-base management]. It enables efficiency, reusability and interoperability” [5].

For an empirically based discipline, still in its formative stage, the theory and framework provided by *Theory of Modeling and Simulation* (1976) provided a sound foundation for M&S to emerge as an established discipline. Such a foundation is necessary to foster the development of M&S-specific methods and the use

of such methods to solve real-world problems faced by practitioners. Even when not fully accepted as gospel (as is theory in Theoretical Physics), the theory is sufficiently mature to provide the necessary skeleton to define a kernel of topics for the emerging M&S discipline and to meaningfully organize these topics. Furthermore, the theory's basic entities and relations such as the separation, and inter-relation, of models and simulators, provide starting points to enumerate and address core M&S research challenges [7]. At the time of first writing of this M&SBoK, theory and practice are being more strongly aligned in a comprehensive formulation of the simulation development life cycle [8]. Indeed, as the richness and applicability of the field increase, it becomes more and more urgent to have an openly available M&SBoK.

1.4 Modeling and Simulation Framework (MSF)

Bernard P. Zeigler

1.4.1 System Concepts

The US Department of Defense M&S Glossary (M&S Glossary) gives these definitions: (1) **system model**: A representation of a system; and (2) **simuland**: The system being simulated by a simulation. These define “system model” and “simuland” in terms of “system” but nowhere to be found is a definition of “system” itself. In contrast, the Modeling and Simulation Framework (MSF), to be reviewed here, includes the “system” as one of the four basic entities along with “experimental frame,” “model,” and “simulator.” Moreover, we need some basic concepts about systems that are reviewed only in outline (for details, see Zeigler et al. [9], *Theory of Modeling and Simulation*, Chap. 1—any edition.)

1.4.1.1 System Specification Hierarchy Levels of System Specification

The MSF sets forth the fundamental entities and relationships in the M&S enterprise. To describe these items, we employ the systems' specification hierarchy as the basis for the MSF.

Table 1.1 identifies five basic levels of system specification forming a System Specification Hierarchy. The fourth column gives an example of a system specification at each level applied to a person in a conversation. Later, we will formulate a conversation, itself, as system composed of two interacting persons. At each level, we know some important things about a system that we did not know at lower levels.

At the lowest level, the *Observation Frame* identifies a portion of the real world (source system) that we wish to model and the means by which we are going to observe it.

Table 1.1 Levels of system specification

Level	Specification name	What we know at this level	Example: a person in a conversation
0	Observation frame	How to stimulate the system with inputs; what variables to measure and how to observe them over a time base	The person has inputs and outputs at the usual cognitive level, such as streams of words
1	I/O behavior	Time-indexed data collected from a source system; consists of input/output pairs	For each input that the person recognizes, the set of possible outputs that the person can produce
2	I/O function	Knowledge of initial state; given an initial state, every input stimulus produces a unique output	Assuming knowledge of the person's initial state when starting the conversation, the unique output response to each input
3	State transition	How states are affected by inputs; given a state and an input what is the state after the input stimulus is over; what output event is generated by a state	How the person transits from state to state under input words and generates output words from the current state
4	Coupled component	Components and how they are coupled together. The components can be specified at lower levels or can even be structure systems themselves—leading to hierarchical structure	A description of a person's I/O behavior in terms of neural components and their interaction by spikes is at this level

As the next two levels, the *I/O Behavior and Function levels*, we have a database of measurements and observations made for the source system. When we get to Level 3, the *State Transition Level*, we have the ability to recreate this data using a more compact representation, such as a formula. Since typically, there are many formulas or other means to generate the same data, the particular means or formula we have settled on constitutes knowledge we did not have at the lower data levels. When people talk about models in the context of simulation studies, they are usually referring to the concepts identified at this level. That is, to them a model means a program to generate data.

At the highest level, the *Coupled Component Level* we have a very specific kind of generative system. In other words, we know how to generate the data observed at Level 1 in a more specific manner in terms of component systems that are interconnected together and whose interaction accounts for the observations made. When people talk about systems, they are often referring to this level of knowledge. They think of reality as being made up of interacting parts so that the whole is the sum (or sometimes claimed, more, or less, than the sum) of its parts. Although some people use the term “subsystems” for these parts, we call them component systems (and reserve the term subsystem for another meaning).

The System Specification Hierarchy is a useful starting point since it provides a unified perspective on what are usually considered to be distinct concepts. From this perspective, there are only three basic kinds of problems dealing with systems and they involve moving between the levels of system knowledge.

In *systems analysis*, we are trying to understand the behavior of an existing or hypothetical system based on its known structure.

Systems inference is done when we do not know what this structure is—so we try to guess this structure from observations that we can make.

Finally, in *systems design*, we are investigating the alternative structures for a completely new system or the redesign of an existing one.

1.4.2 The Entities of the Modeling and Simulation Framework

As illustrated in Fig. 1.1, the basic entities of the framework are *source system*, *model*, *simulator*, and *experimental frame*. The basic inter-relationships among entities are the modeling and the simulation relationships. The entities are defined in Table 1.2 which also characterizes the level of system specification that typically describes the entities. The level of specification is an important feature for distinguishing between the entities, which is often confounded in the literature. You can return to Fig. 1.1 and Table 1.2 to keep an overall view of the framework as we describe each of the components in the following presentation.

1.4.2.1 Source System

The *source system* (we will omit the “source” qualifier, when the context is clear) is the real or virtual environment that we are interested in modeling. It is viewed as a *source of observable data*, in the form of time-indexed trajectories of variables. The

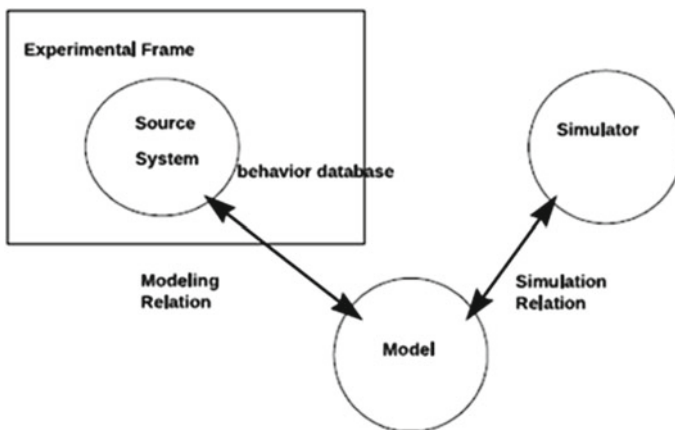


Fig. 1.1 Fundamental entities and relationships in the M&S framework

Table 1.2 Defining the basic entities in M&S and their usual levels of specification

Basic entity	Definition	Related system specification levels
Source system	Real or artificial source of data	Known at level 0
Behavior database	Collection of gathered data	Observed at level 1
Experimental frame	Specifies the conditions under which system is observed or experimented with	Constructed at levels 3 and 4
Model	Instructions for generating data	Constructed at levels 3 and 4
Simulator	Computational device for generating behavior of the model	Constructed at level 4

data that has been gathered from observing or otherwise experimenting with a system is called the *system behavior database*. As indicated in Table 1.2, this concept of system is a specification at level 0 and its database is a specification at level 1. This data is viewed or acquired through experimental frames of interest to the modeler.

Applications of M&S differ with regard to how much data is available to populate the system database. In *data-rich* environments, such data is abundant from prior experimentation or can easily be obtained from measurements. In contrast, *data-poor* environments offer meager amounts of historical data or low-quality data (whose representativeness of the system of interest is questionable). In some cases, it is impossible to acquire better data (e.g., of combat in real warfare); in others, it is expensive to do so (e.g., topography and vegetation of a forest). In the latter case, the modeling process can direct the acquisition of data to those areas that have the highest impact on the final outcome.

1.4.2.2 Experimental Frame

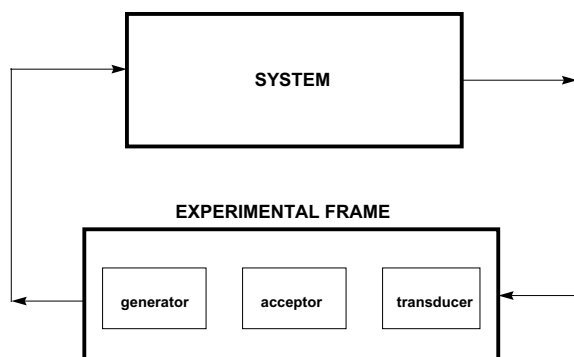
An *experimental frame* is a specification of the conditions under which the system is observed or experimented with. As such, an experimental frame is the operational formulation of the objectives that motivate a modeling and simulation project. For example, out of the multitude of variables that relate to a forest, the set {lightning, rain, wind, smoke} represents one particular choice. Such an experimental frame is motivated by the interest in modeling the way lightning ignites a forest fire. A more refined experimental frame would add the moisture content of the vegetation and the amount of unburned material as variables. Thus, many experimental frames can be formulated for the same system (both source system and model) and the same experimental frame may apply to many systems. Why would we want to define many frames for the same system? Or apply the same frame to many systems? For the same reason, we might have different objectives in modeling the same system or have the same objective in modeling different systems. More of this in the sequel.

There are two equally valid views of an experimental frame. One views a frame as a definition of the type of data elements that will go into the database. The second views a frame as a system that interacts with the system of interest to obtain the data of interest under specified conditions. In this view, the frame is characterized by its implementation as a measurement system or observer. In this implementation, a frame typically has three types of components (as shown in Fig. 1.2): *generator* that generates input segments to the system; *acceptor* that monitors an experiment to see the desired experimental conditions are met; and *transducer* that observes and analyzes the system output segments.

1.4.2.3 Objectives and Experimental Frames

Objectives for modeling relate to the role of the model in systems design, management, or control. The statement of objectives serves to focus model construction on particular issues. Figure 1.3 depicts the process of transforming objectives into experimental frames. Typically, modeling objectives concern system design. Here, measures of the effectiveness of a system in accomplishing its goal are required to evaluate the design alternatives. We call such measures, *outcome measures*. In order to compute such measures, the model must include variables, we will call *output variables*, whose values are computed during execution runs of the model. The mapping of the output variables into outcome measures is performed by the transducer component of the experimental frame. Often there may be more than one layer of variables intervening between output variables and outcome measures. For example, in military simulations, *measures of performance* are output variables that typically judge how well parts of a system are operating. For example, the success of a missile in hitting its target is a performance measure. Such measures enter as factors into outcome measures, often called *measures of effectiveness*, that measure how well the overall system goals are being achieved, e.g., how many battles are actually won by a particular combination of weapons, platforms, personnel, etc. The implication is that high performing components are necessary, but not sufficient, for highly effective systems, in which they must be coordinated together to achieve the overall goals.

Fig. 1.2 Experimental frame and its components



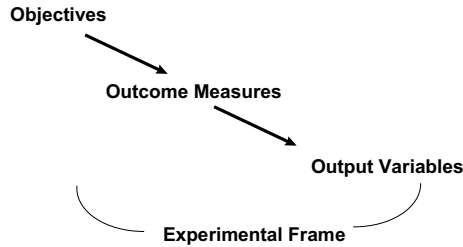


Fig. 1.3 Transforming objectives to experimental frames

Example: Two-person Interaction

The conversation example for the System Specification Hierarchy of Table 1.1 actually assumes an underlying experimental frame that was not completely specified. We can specify the objective of trying to characterize when a two-person interaction is a valid conversation, i.e., when the participants exchange words that make sense to an observer. We restrict the interaction to an exchange of greetings of two people passing by each other. There are relatively few pairs of words that make sense such as “Hello, Hi” as a greeting by one person and a response of the other, whereas most other pairs of words do not. Such pairs are in effect a description at the I/O Behavior level for the interaction of two persons. An experimental frame in this case centers on collecting such I/O pairs in both a real two-person encounter and a simulation model of it. The acceptor component of such a frame can monitor the interaction for pairs judged to be indicative of a valid conversation.

Model

In its most general guise, a *model* is a system specification at any of the levels of the System Specification Hierarchy. However, in the traditional context of M&S, the system specification is usually done at levels 3 and 4. Thus, the most common concept of a simulation model is that it is a *set of instructions, rules, equations, or constraints for generating I/O behavior*. In other words, we write a model with a state transition and output generation mechanisms (level 3) to accept input trajectories and generate output trajectories depending on its initial state setting. Such models form the basic components in more complex models that are constructed by coupling them together to form a level 4 specification.

Example: Two-person Interaction

An example of a conversation between two persons can be modeled as agents interacting through exchange of messages carried by discrete events. This constitutes a coupled model with atomic model components. Each component alternates between speaking and listening phases. If moreover, the components adhere to the discipline that only one is in the speaking phase at any time, then the result represents a valid conversation. If at any time, both components are in the same phase (speaking or listening) then the experimental frame acceptor just discussed will stop the simulation and declare this as an invalid conversation.

There are many meanings that are ascribed to the word “model.” For example, a model is conceived as any physical, mathematical, or logical representation of a system, entity, phenomenon, or process. *The definition in terms of system specifications has the advantages that it has a sound mathematical foundation and it has a definite semantics that everyone can understand in unambiguous fashion.* Like other formal definitions, it cannot capture all meanings in the dictionary. However, it is intended to capture the most useful concepts in the M&S context.

1.4.2.4 Simulator

As a set of instructions, a model needs some agent capable of actually obeying the instructions and generating behavior. We call such an agent a *simulator*. Thus, a simulator is any computation system (such as a single processor, a processor network, the human mind, or more abstractly an algorithm), capable of executing a model to generate its behavior. A simulator is typically specified at a high level since it is a system that we design intentionally to be synthesized from components that are off-the-shelf and well-understood. Separating the model and simulator concepts provides a number of benefits for the framework:

- The same model, expressed in a formalism, may be executed by different simulators thus opening the way for portability and interoperability at a high level of abstraction.
- Simulator algorithms for the various formalisms may be formulated and their correctness rigorously established.
- The resources required to correctly simulate a model afford a measure of its complexity.

Example: Two-person Interaction

The two-person discrete event model just mentioned can be formulated within the Discrete Event System Specification (DEVS) formalism as illustrated in Chap. 10 of (Zeigler et al. 2017). This then allows the model behavior to be generated by a DEVS simulator, i.e., a simulation program implementing the rules of the Abstract DEVS simulation protocol that guarantees correct simulation of any DEVS model.

1.4.3 Primary Relations Among Entities

The entities—system, experimental frame, model, and simulator—become truly significant only when properly related to each other. For example, we build a model of a particular system for some objective—only some models, and not others, are suitable. Thus, it is critical to the success of a simulation modeling effort that certain relationships hold. The two most fundamental are the modeling and the simulation relations (Table 1.3).

Table 1.3 Primary relationships among entities

Basic relationship	Definition	Related system specification levels
Modeling relation Replicative validity Predictive validity Structural validity	Concerned with how well model-generated behavior agrees with observed system behavior	Comparison is at level 1 Comparison is at level 2 Comparison is at level 3, 4
Simulation relation Correctness	Concerned with assuring that the simulator carries out correctly the model instructions	Basic comparison is at level 2; involves homomorphism at levels 3 or 4

1.4.3.1 Modeling Relation: Validity

The basic modeling relation, *validity*, refers to the relation between a model, a system, and an experimental frame. Validity is often thought of as the degree to which a model faithfully represents its system counterpart. However, it makes much more practical sense to require that the model faithfully captures the system behavior only to the extent demanded by the objectives of the simulation study. In the MSF, the concept of validity answers the question of whether it is impossible to distinguish the model and system in **the experimental frame of interest**.

The most basic concept, *replicative validity*, is affirmed if, for all the experiments possible within the experimental frame, the behavior of the model and system agree within acceptable tolerance. Thus, replicative validity requires that the model and system agree at the I/O relation level 1 of the system specification hierarchy.

Stronger forms of validity are *predictive validity* and *structural validity*. In predictive validity, we require not only replicative validity, but also the ability to predict as yet unseen system behavior. To do this, the model needs to be set in a state corresponding to that of the system. Thus, predictive validity requires agreement at the next level of the system hierarchy, that of the I/O function level 2. Finally, *structural validity* requires agreement at level 3 (state transition) or higher (coupled component). This means that the model not only is capable of replicating the data observed from the system but also mimics in step-by-step, component-by-component fashion, the way that the system does its transitions.

The term *accuracy* is often used in place of validity. Another term, *fidelity*, is often used for a combination of both validity and detail. Thus, a high-fidelity model may refer to a model that is both high in detail and in validity (in some understood experimental frame). However, when used this way, beware that there may be a tacit assumption that high detail alone is needed for high fidelity, as if validity is a necessary consequence of high detail. In fact, it is possible to have a very detailed model that is nevertheless very much in error, simply because some of the highly resolved components function in a different manner than their real system counterparts.

1.4.3.2 Simulation Relation: Simulator Correctness

The basic *simulation* relation, *simulator correctness*, is a relation between a *simulator* and a *model*. A *simulator correctly simulates a model* if it is guaranteed to faithfully generate the model's output trajectory given its initial state and its input trajectory. Thus, simulator correctness requires agreement at the I/O function level 2. In practice, simulators are constructed to execute not just one model but a family of possible models. This flexibility is necessary if the simulator is to be applicable to a range of applications. In such cases, we must establish that a simulator will correctly execute a particular class of models. Since the structures of both the simulator and the model are at hand, it may be possible to prove correctness by showing that a *homomorphism* relation holds. Here, a homomorphism is a correspondence between simulator and model states that is preserved under transitions and outputs.

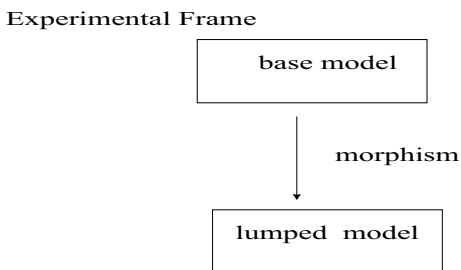
1.4.4 Other Important Relationships

Besides the two fundamental modeling and simulation relationships, there are others that are important for the M&SBoK. These relations have to do with the interplay and comparative complexity-related orderings of models and experimental frames.

1.4.4.1 Modeling as Valid Simplification

The inescapable fact about modeling is that it is severely constrained by complexity limitations. Complexity is at heart, an intuitive concept—the feeling of frustration or awe that we all sense when things get too numerous, diverse, or intricately related to discern a pattern, to see all at once—in a word, to comprehend. Generalizing from the boggled human mind to the overstressed simulator suggests that the *complexity of model* can be measured by the resources required by a particular simulator to correctly interpret it. As such, complexity is measured relative to a particular simulator, or class of simulators. However, properties intrinsic to the model are often strongly correlated with complexity independently of the underlying simulator. Successful modeling can then be seen as *valid simplification*. We need to *simplify*, or reduce the complexity, to enable our models to be executed on our resource-limited simulators. But the simplified model must also be *valid*, at some level, and within some experimental frame of interest. As in Fig. 1.4, there is always a pair of models involved, call them the *base* and *lumped* models. Here, the base model is typically “more capable” and requires more resources for interpretation than the lumped model. By the term “more capable,” we mean that the base model is valid within a larger set of experimental frames (with respect to a real system) than the lumped model. However, the important point is that within a **particular frame of interest** the lumped model might be just as valid as the base model. The concept of morphism affords criteria for judging the equivalence of base and lumped models with respect to an experimental frame.

Fig. 1.4 Base/lumped model equivalence in experimental frame



1.4.4.2 Experimental Frame—Model Relationships

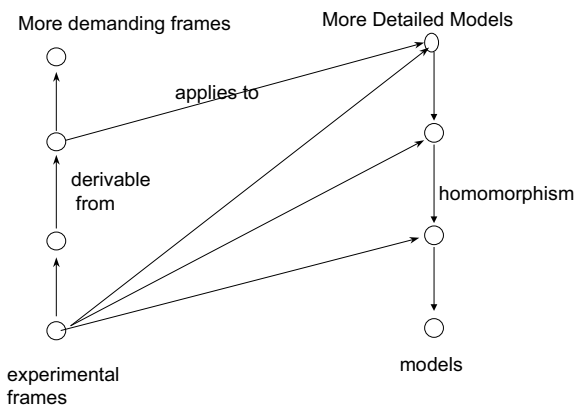
Assume that we have a whole repository of models and experimental frames that have been built up over years of experience. Then, it is critical to have an ability to ask whether there are any experimental frames that meet our current objectives and whether there are models that can work within this frame. Only those models have a chance of providing valid answers to our current questions. The relation that determines if a frame can logically be applied to a model is called *applicability* and its converse is called *accommodation* (Table 1.4). Notice that validity of a model in a particular experimental frame requires, as a precondition, that the model accommodates the frame.

The degree to which one experimental frame is more restrictive in its conditions than another is formulated in the *derivability* relation. A more restrictive frame leaves less room for experimentation or observation than one from which it is derivable. So, as illustrated in Fig. 1.5, it is easier to find a model that is valid in a restrictive frame for a given system. It turns out that applicability may be reduced to derivability. To see this, define the *scope* frame of the model to represent the most relaxed conditions under which it can be experimented with (this is clearly a characteristic of the model.) Then, a frame is applicable to a model, if it is derivable from the scope frame of the model. This means that a repository need not support both applicability and derivability queries. Only the latter is sufficient if each model has an associated scope frame.

Table 1.4 Other M&S relationships important when dealing with a model repository

Relationship	Definition
Experimental frame applies to a model (or “is applicable to”)	The conditions on experimentation required by the frame can be enforced in the model
Model accommodates experimental frame	Frame is applicable to the model
Experimental frame 1 is derivable from experimental frame 2	Any model that accommodates experimental frame 2 also accommodates experimental frame 1

Fig. 1.5 Illustrating important M&S relations relevant to model repositories



1.4.5 Time

A *time base* is an ordered set, usually the real numbers, for indexing events that models the flow of actual time (Table 1.5). If the interpretation of such a time base is left abstract in this manner, we refer to it as *logical* time. In contrast, when we consider events happening in the real world, in real time, we refer to a time variable as measured by an actual clock. Thus, *physical* time, also called metric time or wall-clock time, is measured by ticks of physical clocks, while logical time is measured by ticks of a clock somehow embedded in a model. Also, as relativity theory made clear, time, as perceived by observers at different locations may be different. Based on this distinction, time can be either *local* or *global*. The former is valid only within a component of a system; the latter is valid in the whole system. Thus, there are at least two dimensions for classifying time: one along the logical/physical axis and the other along the local/global axis. Consequently, a time base can be interpreted as falling in any one of the four combinations shown in 0.

Traditionally, modeling and simulation have considered mainly the first (global, logical) combination. That is, we assume all components of a modeled system have the same time frame of reference and we consider time as an abstract quantity.

Table 1.5 A time taxonomy

		Logical/physical	
		Logical time	Physical time
Local/global	Global time	Global logical: All components operate on the same abstract time base All components operate on the same abstract time base	Global, physical: All components operate on the same system clock
	Local time	Local, logical: A component operates on its own abstract time base	Local, physical: A component operates on its own system clock

However, when a model is executing in a simulator, which may be distributed among computer nodes in a network and may also be interacting with the real world, it is hard to maintain this fiction. We note that synchronization between time bases requires maintaining a correspondence between the two. For example, a distributed simulation protocol synchronizes the local, logical times maintained by the individual simulator nodes. Another example of synchronization occurs in a real time, human-in-the-loop simulation-based training. Here, the simulator employs a physical time base (e.g., computer system clock) to synchronize between a pilot's physically perceived time base and the logical time of a model of the aircraft being simulated.

1.4.6 Mapping Informal Terminology to MSF Formalization

Bernard P. Zeigler

Often in the literature of M&S, terminology is defined conceptually but not in the mathematically precise manner of the MSF presented here. For example, we have seen how the US Department of Defense M&S Glossary (M&S Glossary—M&SCO) defines “system model” and “simuland” in terms of “system” without giving a definition of “system” itself. A rough equivalence between terminology often found in the literature and entities of the framework can be established in Tables 1.6 and 1.7:

Since the MSF defines its entities as mathematical systems, it can define typical activities involved in M&S work as mathematical relations. A rough equivalence is given in Table 1.7:

Table 1.6 Some common conceptual definitions and MSF equivalents

Conceptual definition of object	MSF formalization
A simuland is the real-world system of interest. It is the object, process, or phenomena to be simulated	Real-world system is a source of data that can be represented by a system specification at a behavioral level
A model is a representation of a simuland, broadly grouped into conceptual and executable types	Model is a set of rules for generating behavior and can be represented by a system specification at a structural level. A modeling formalism, e.g., DEVS, enables conceptual specification and is mapped to a simulation language for execution by a simulator
Simulation is the process of executing a model over time	A simulator is a system capable of generating the behavior of a model; simulators come in classes corresponding to formalisms, e.g., an abstract DEVS simulator describes implementable simulators of DEVS models
The results of simulation are the output produced by a model during a simulation	The behavior of a model generated by a simulator constitutes a specification at the behavior level

Table 1.7 Conceptual definitions and MSF equivalents

Conceptual definition of activity	MSF formalization
Verification is the process of determining if an implemented model is consistent with its specification	There is a relation, called simulation correctness, between models and simulators. Verification is the process of proving this correctness in a simulator generating the behavior of the model. When this is done for a formalism, it certifies a simulator as correct for any model of the associated class
Validation is the process of determining if a model behaves with satisfactory accuracy consistent with the study objectives within its domain of applicability to the simuland it represents	There is a relation, called validity in a frame, between models and real systems within an experimental frame. Validation is the process of establishing that the behaviors of the model and real system agree in the frame in question. The frame can capture the intended objectives (extended to intended uses), domain of applicability, and accuracy requirements
Abstraction is the omission or reduction of detail not considered necessary in a model	Abstraction is the process of constructing a lumped model from a base model intended to be valid for the real system in a given experimental frame

The definition of validation is a synthesis of various definitions in the literature that separately relate the model to a real system, the purposes of model construction, the domain of applicability, and the accuracy required. For example, Balci [10] defines validation as the assessment of behavioral or representational accuracy and then later conditions accuracy on intended use. Our intent is to best represent the conceptual literature for the purposes of relating it to the MSF.

1.5 Basic System Entity Structure (SES) Concepts

Thorsten Pawletta

The *System Entity Structure (SES)* is a structural knowledge representation scheme introduced by Zeigler [11]. It contains knowledge of decomposition, taxonomy, and coupling of a system. In combination with a *Model Base (MB)*, it supports different concepts for system modeling, investigating design alternatives, reusing good designs, and collaborative modeling [7, 12, 13].

Figure 1.6 shows the general procedure model of an SES/MB-based M&S according to Pawletta et al. [14]. Possible configurations of a system or a family of systems are analyzed. That means, basic dynamic components, their relations, and parameter settings are identified. Dynamic components are modeled or implemented as reusable basic systems with defined input and output interfaces and organized in a MB. The possible system structures and parameter settings are modeled with an SES, which specifies formal links to basic systems in the MB. Figure 1.7 shows an

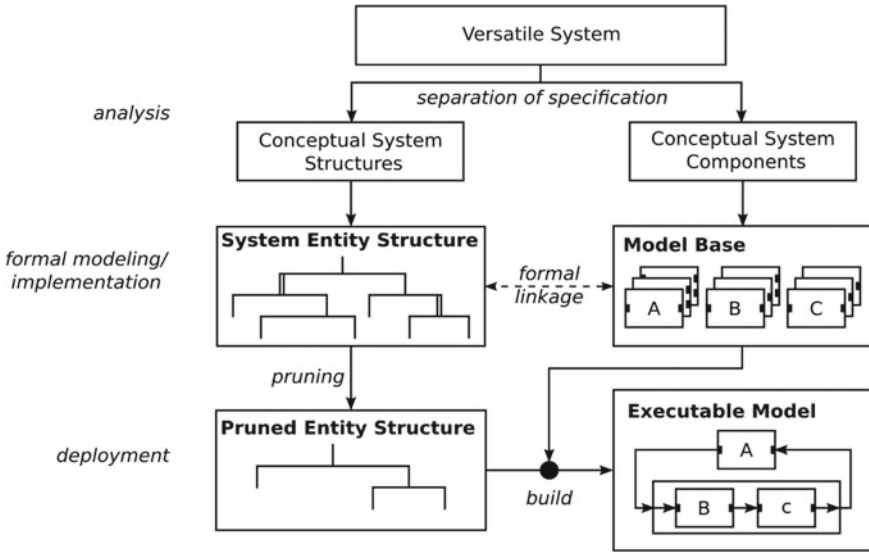


Fig. 1.6 Procedure model of an SES/MB-based M&S according to Pawletta et al. [14]

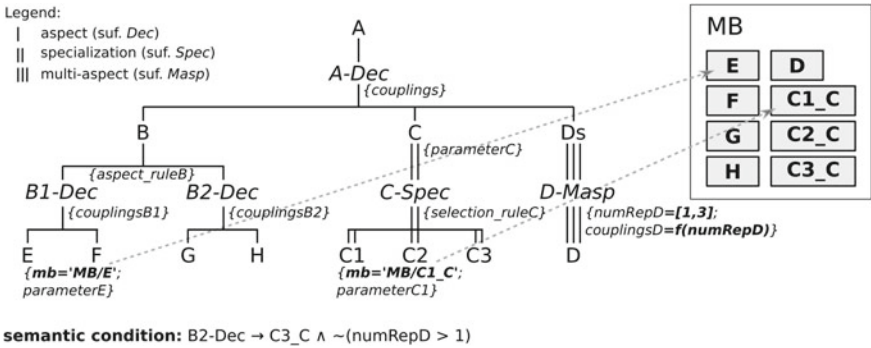


Fig. 1.7 Basic example of a SES, which describes ten admissible system structures. There are two variants of composing entity *B* (aspects *B1-Dec* and *B2-Dec*), three variants for selecting a specific type of entity *C* and entity *D* can be replicated one, two, or three times. However, the *semantic condition* limits the selection to *C3_C* and only one replication of *D* when using aspect *B2-Dec*. The coupling relations of the replicated entities *D* result dynamically dependent on the value of attribute *numRepD*

example of an SES with associated MB. In the application phase, executable models are generated with transformation methods such as *pruning* and *build*. Based on defined objectives, the pruning method derives a unique system configuration from the set of possible configurations. The result of pruning is called *Pruned Entity Structure (PES)*. Based on the information in the PES, the build method generates an *Executable Model* using basic systems from the MB.

The transformation methods can be executed interactively or automatically. Extended SES/MB-based architectures for automated and reactive pruning of SES are presented in Schmidt et al. [15] and Folkerts et al. [16]. Different approaches of the build method are discussed in Folkerts et al. [17]. The generation of executable models for different simulators based on a uniform MB is discussed by Pawletta and Folkerts (Sect. 18.6).

An SES is represented by a directed tree structure, as illustrated in Fig. 1.7. The different edges are related to different node types. Each node can define *attached variables*, also called *attributes*. Real-world or imaginary objects are represented by *entity nodes*. Entity attributes represent properties of the respective object. The root and the leaves are always entities. Relations between entities are specified by three types of *descriptive nodes*, called *aspect*, *multi-aspect*, and *specialization*. The attributes of descriptive nodes specify relations between their parent node and children nodes or decisions for the pruning process. *Aspects* describe how entities can be decomposed in partial entities. Coupling relations can be specified in a *couplings* attribute. *Multi-aspects* describe the decomposition of an entity into entities of the same class. They define an additional attribute, called *Number of Replications* (*numRep*). The taxonomy of an entity is described by *specialization(s)* and concerns admissible variants of an entity. Rules for selecting a variant during pruning can be defined in a *selection rule* attribute. With the extended procedural knowledge representation according to Pawletta et al. [14], attributes can be dynamically assigned values. For example, coupling relations of a *multi-aspect* can result dependent on the value of attribute *numRep*.

The semantics of the SES is defined by axioms. Types of each node have to follow the axiom *alternating mode*. Every entity node has to be followed by a descriptive node, and vice versa. A *strict hierarchy* is needed. In every path of the tree, a name of a node may occur only once. If nodes in different paths have the same name, they need to have the same variables and isomorphic partial trees. This is called *uniformity*. Nodes on the same level of a hierarchy, called sibling nodes, have to be *valid brothers*, meaning that sibling nodes must not have the same name. The axiom of *attached variables* implies that a node must not have variables of the same name. The axiom of *inheritance* implies that during pruning, the parent and the child of a specialization combine their variables, aspects, and specializations. The configurations modeled in an SES tree can be delimited by *selection constraints* and *semantic conditions*.

There are numerous additional SES concepts, such as using the SES as a general ontology for data modeling [18], the specification of abstraction hierarchies and time granularities for families of systems [19], interfaces for automated, reactive pruning [15], methods for the pruning of deep hierarchies of certain node type combinations [16], or the combination with performance metrics to evaluate and select the best possible system configurations [20].

References











1. Ören TI (2011a) The many facets of simulation through a collection of about 100 definitions. *SCS M&S Magazine*, 2:2 (April), pp. 82–92. <http://scs.org/wp-content/uploads/2016/12/2011-04-Issue06-6.pdf>. Accessed: 2019–01–24
2. Ören TI (2011b) A critical review of definitions and about 400 types of modelling and simulation. *SCS M&S Mag* 2:3 (July):142–151. <http://scs.org/wp-content/uploads/2016/12/2011-04-Issue06-6.pdf>. Accessed: 2019–01–24
3. Ören T, Mittal S, Durak U (Invited Chapter) Modeling and simulation: the essence and increasing importance. In: Niazi MA (ed) *Modeling and simulation of complex communication networks*, the IET book series on big data. (Appendix A: A list of over 750 types of simulation, Appendix B: A list of 900 types of models, and Appendix C: A list of 120 types of input)
4. Appendix A1—Dictionaries of modeling and simulation, cybernetics, and systems (compiled by Tuncer Ören)
5. Durak U, Ören T, Tolak A (2017) An index to the body of knowledge of simulation systems engineering. In: Tolak A, Ören T (eds) *The profession of modeling and simulation: discipline, ethics, education, vocation, societies, and economics*. Wiley.
6. Sargent RG (2017) A perspective on fifty-five years of the evolution of scientific respect for simulation. In: Chan WKV, D'Ambrogio A, Zacharewicz G, Mustafee N, Wainer G, Page E (eds) *Page proceedings of the 2017 winter simulation conference*
7. Zeigler BP, Muzy A, Kofman E (2018) *Theory of modelling and simulation*, 3rd edn. Elsevier Academic Press
8. Zhang L, Zeigler BP, LaiLi Y (2019) *Model engineering for simulation*, (co-volume with TMS3rd Ed.) Elsevier
9. Zeigler BP, Muzy A, Kofman E (3rd edn, 2019) *Theory of modeling and simulation: discrete event & iterative system computational foundations*, Elsevier. Please note: Chapters from this latest (3rd) edition can be downloaded from: <https://www.sciencedirect.com/book/9780128133705/theory-of-modeling-and-simulation> or from Researchgate.net (requesting from Alex Muzy or B.P. Zeigler)
10. Balci O (2012) A life cycle for modeling and simulation. *Simulation* 88(7):870–883
11. Zeigler BP (1984) *Multifaceted modelling and discrete event simulation*. Academic Press
12. Rozenblit JW, Zeigler BP (1993) Representing and constructing system specifications using the system entity structure concepts. In: *Winter simulation conference*, Los Angeles, pp 604–611
13. Zeigler BP, Praehofer H, Kim TG (2000) *Theory of modelling and simulation*, 2nd edn. Elsevier Academic Press
14. Pawletta T, Durak U, Schmidt A (2019) Modeling and simulation of versatile technical systems using an extended system entity structure/model base infrastructure. In: Zhang L, Zeigler BP, Laili Y (eds) *Model engineering for simulation*. Elsevier Academic Press, pp 393–418 (Chapter 18)
15. Schmidt A, Durak U, Pawletta T (2016) Model-based testing methodology using system entity structures for MATLAB/simulink models. *Simul Trans Soc Model Simul Int* 92 (8):729–746
16. Folkerts H, Pawletta T, Deatcu C, Zeigler BP (2020) Automated, reactive pruning of system entity structures for simulation engineering. *SpringSim'20*, May 19–21, Fairfax, VA, USA, Society for Modeling & Simulation International (SCS), 12 pages
17. Folkerts H, Pawletta T, Deatcu C (2020) Model generation for multiple simulators using SES/MB and FMI. *ASIM SST 2020*, October 14–15, St. Augustin, Germany, ARGESIM report 59, pp 13–20. <https://doi.org/10.11128/arep.59.a59003>
18. Zeigler BP, Hammonds P (2007) *Modeling and simulation-based data engineering*. Academic Press, Burlington

19. Santucci JF, Capocchi L, Zeigler BP (2016) System entity structure extension to integrate abstraction hierarchies and time granularity into DEVS modeling and simulation. *Simul Trans Soc Model Simul Int* 92(8):747–794
20. Capocchi L, Santucci JF, Pawletta T, Folkerts H, Zeigler BP (2020) Discrete-event simulation model generation based on activity metrics. *Simul Model Pract Theory* 103:102122 (41 pages). <https://doi.org/10.1016/j.simpat.2020.102122>
21. Fujimoto R, Bock C, Chen W, Page E, Panchal JH (eds) (2017) *Research challenges in modeling and simulation for engineering complex systems*. Springer International Publishing AG
22. Zeigler BP (1976) *Theory of modeling and simulation*. Academic Press, Elsevier



M&S Bok Core Areas and the Big Picture

2

Tuncer Ören , Umang Kant , Mayank Sing , Paul Fishwick , Mamadou Kaba Traoré , Lin Zhang , Yuanjun Laili , Bernard P. Zeigler , Andreas Tolk , Gregory Zacharewicz , Mohammad S. Obaidat, and Balqies Sadoun

Abstract

This chapter deals with models, data, and their relations and use. It introduces the big picture of the SCS M&S Body of Knowledge, providing taxonomies for models and their uses and resulting perceptions as well as an extensive section on data. It addresses the questions of modeling and various modeling formalisms, model engineering, and model curation for repository integration. The chapter concludes with model-based simulation approaches and a section on the transient elimination in simulation.

T. Ören (✉)

University of Ottawa, Ottawa, ON, Canada
e-mail: toren@uottawa.ca

U. Kant

Delhi Technological University, New Delhi, India

M. Sing

Consilio Research Lab, Noida, India
e-mail: dr.mayank.singh@iecee.org

P. Fishwick

University of Texas at Dallas, Richardson, TX, USA
e-mail: Paul.Fishwick@utdallas.edu

M. K. Traoré

University of Bordeaux, Bordeaux, France
e-mail: mamadou-kaba.traore@u-bordeaux.fr

L. Zhang · Y. Laili

Beihang University, Beijing, China
e-mail: johnlin9999@163.com

Keywords

Modeling and simulation • Simulation data • Modeling formalisms • Model engineering • Model curation • Transient elimination

2.1 The Big Picture

Tuncer Ören

This section is adopted from [1, 2] and has been updated accordingly. Simulation is a model-based activity [3]. Hence, core areas of simulation include models and data; their relationship is straightforward. Data is used to formulate and calibrate models, and models are used to generate data. Models are formulated according to modeling formalisms. Model engineering covers all aspect of formulation, processing, and use of models. Experiments and experience are the main reasons of using modeling and simulation.

Models or representations of existing or non-existing reality are used in simulation. Study of reality/model dichotomy can help us explore different types of simulation. Table 2.1 outlines model/reality dichotomy based on usage in art, engineering, science, decision support, education, training, entertainment, as well as pretense and representation.

Y. Laili
e-mail: lailiyuanjun@buaa.edu.cn

B. P. Zeigler
University of Arizona, Tucson, USA
e-mail: zeigler@rtsync.com

A. Tolk
The MITRE Corporation, Charlottesville, VA, USA
e-mail: atolk@mitre.org

G. Zacharewicz
IMT-Mines Alès, 6 avenue de Clavières, 30100 Ales, France
e-mail: Gregory.Zacharewicz@mines-ales.fr

M. S. Obaidat
University of Texas-Permian Basin, Odessa, TX 79762, USA

King Abdullah II School of Information Technology, University of Jordan, Jordan and
University of Science and Technology, Amman, Jordan

B. Sadoun
College of Engineering, Al Balqa' Applied University, Al Salt, Jordan

Table 2.1 Model/reality dichotomies

Based on reality/model dichotomy				
In simulation, models or representations of existing or non-existing reality are used. Study of reality/model dichotomy can help us explore different types of simulation. Reality/model dichotomy depends on the purpose: art, engineering, science, decision support, education, training, entertainment, as well as pretense, representation				
Art	In art, reality is a source of inspiration and is called ‘model’ The model—for an artist—is what a simulationist would say real system!			
Engineering	There are two possibilities for design and control problems:			
	<table border="1"> <tr> <td>Design</td> <td>A design (or a model) is an instrument to engineer a system</td> </tr> <tr> <td>Control</td> <td>A model is a basis to control a system</td> </tr> </table>	Design	A design (or a model) is an instrument to engineer a system	Control
Design	A design (or a model) is an instrument to engineer a system			
Control	A model is a basis to control a system			
Science	Use of models in analysis problems			
	Analysis A model is a representation to understand a system			
Decision support	A model is a substitute of reality to perform experiments A model or a representation of reality can be used to gain experience to develop/enhance three types of skills			
Education	A model is a representation to explain/teach dynamic systems			
Training	A representation of a system provides experience to enhance three types of skills – Motor skills (virtual simulation, simulators, virtual simulators) – Decision-making skills (constructive simulation; serious games) – Operational skills (live simulation)			
Entertainment	A representation of a system provides experience for entertainment			
pretense/representation	We are often exposed to simulated reality, in postmodern societies [4] (boundary becomes blurred)			

Simulation can be **perceived from different perspectives** such as purpose of use, problem to be solved, connectivity of operations, types of knowledge processing, and philosophy of science.

Table 2.2 outlines perceptions of simulation based on purpose of use and problem to be solved.

Use of simulation experiments for decision support has many aspects. Table 2.3 outlines several ways simulation experiments are used for decision support.

Perceptions of simulation based on the connectivity of operations of simulation and system of interest are outlined in Table 2.4.

Based on knowledge processing, simulation can be perceived as a computational activity, a system theory-based activity, a model-based activity, a knowledge generation activity, and a knowledge-processing activity. Table 2.5 outlines perceptions of simulation from the perspectives of computational and system theory-based activities.

As a model-based activity, simulation has several advantages and covers four types of activities. Table 2.6 outlines advantages and four types of activities.

Table 2.2 Perceptions of simulation based on purpose of use and problem to be solved

Based on purpose of use	
This perspective is user oriented and clarifies main categories of uses of simulation	
Experiment	Simulation is <i>performing experiments</i> (for decision support, understanding, and education)
Experience	Simulation <i>provides experience</i> (under controlled conditions) for: Training (for gaining/enhancing competence) for three types of skills: <ul style="list-style-type: none"> – Motor skills (virtual simulation by using virtual or simulated equipment) – Decision and/or communication skills (constructive simulation—serious games) – Operational skills (live simulation) Entertainment (gaming simulation)
Imitation	Imitation is another aspect of simulation (such as simulated leather or fake) and is the original meaning attached to the term simulation. Art can also be perceived as experience [5]. However, these aspects are beyond the scope of this document
Based on problem to be solved	
(Simulation is perceived as an infrastructure to support real-world activities and is perceived as not being the “real thing”)	

Table 2.3 Several ways simulation experiments are used for decision support

Use of simulation experiments for decision support
Prediction of behavior and/or performance of the system of interest within the constraints inherent in the simulation model (e.g., its granularity) and the experimental conditions
Evaluation of alternative models, parameters, experimental and/or operating conditions on model behavior or performance
Sensitivity analysis of behavior or performance of the system of interest based on granularities of different models, parameters, experimental and/or operating conditions
Evaluation of behavior and/or performance of engineering designs
Virtual prototyping
Testing
Planning
Acquisition (or simulation-based acquisition)
Proof of concept

Model analysis covers many model-based activities. It can be descriptive model analysis or model characterization and evaluative model analysis or model evaluation. Table 2.7 outlines types of model characterization.

Model evaluation or evaluative model analysis can be done with respect to modeling formalisms, another model (model comparison), real system, and goal of study. Table 2.8 outlines types of model evaluations.

Model transformation is another type of model-based activity. Table 2.9 outlines types of model transformations.

Table 2.4 Perceptions of simulation based on the connectivity of operations of simulation and system of interest

Based on connectivity of operations of simulation and system of interest	
Two possibilities can be identified	
No connection	This is the case of <i>stand-alone simulation</i> which is the most widely used type of simulation
Operations are interwoven	This is the case of <i>integrated</i> or <i>symbiotic simulation</i> . It can be used for two purposes
	To enrich real system's operations (The system of interest and the simulation program operate simultaneously) for <ul style="list-style-type: none"> – Online diagnostics (or simulation-based diagnostics) – Simulation-based augmented/enhanced reality operation (for training to gain/enhance motor skills and related decision-making skills) (e.g., AI airplane in a dogfight training with real aircrafts)
	To support real system operations (The system of interest and the simulation program operate alternately to provide simulation-based predictive displays) <ul style="list-style-type: none"> – Parallel experiments while system is running

Table 2.5 Perceptions of simulation from the perspectives of computational and system theory-based activities

Based on types of knowledge processing	
Simulation is a <i>computational activity</i> , a <i>system theory-based activity</i> , a model-based activity, a knowledge generation activity, and a knowledge-processing activity	
<i>Computational activity</i>	This point of view clarifies some legacy definitions of simulation Simulation is “A method for implementing a model over time.” [6] “ Simulation: <i>the exercising of a model over time.</i> ” [7] Some examples of circular, incorrect, and misleading definitions of simulation were cited by Ören [1]
<i>System theory-based activity</i>	This approach [8] is the basis of system-theoretic approach for modeling and symbolic model processing <ul style="list-style-type: none"> – The widely used standard approach is discrete event system specification (DEVS) [9] – General system theory implementor (GEST) was a system theory-based [10] declarative language for continuous systems expressible by ordinary differential equations [11, 12]. Its coupling features, which include time-varying coupling [13], go beyond traditional programming [14]. However, there is no commercial version – Dynamo (DYNAmic models) is a historic language developed by J. W. Forrester et al. in late 1950s and was based in industrial dynamics (later system dynamics) (Wikipedia-Dynamo)

Table 2.6 Advantages of model-based simulation and four types of activities

Based on types of knowledge processing : simulation is a <i>model-based activity</i>
Some of the advantages of model-based approach [3] are efficiency in computerization, reliability, reusability and composability, and interoperability
– Efficiency in computerization: model bases (or model repositories) may contain model specifications that can easily be converted into programs. Hence, programming aspect can and should be fully automated. This aspect also eliminates programming errors and contributes to the reliability of the computerization of models
– Reliability: models can easily be read and understood by specialists in the field. This aspect can help assuring model reliability
Model specifications can be checked by specialized software as well as manually for consistency, completeness, and correctness. This aspect is definitely superior to traditional V&V techniques that work on code only and can be the basis for built-in reliability in M&S studies
– Reusability and composability: model specifications can easily be modified for model reusability as well as model composition. Some of the model composability techniques can be dynamically applicable for systems that not only have dynamic behavior but also can and should be modified dynamically as the simulation evolves
– Interoperability: it is highly desirable to check interoperability of model specifications rather than the codes of models since executability of code does not necessarily signify its semantic interoperability
As a model-based activity, simulation includes the following four types of activities : model building, model-based management, parameter base management, and model processing
– Model building It consists of modeling (from scratch) and model composition including dynamic model composition at run time (or execution time)
– Model-based management It includes model search, semantic model search, and model integrity
– Parameter base management In some applications, such as nuclear fuel waste management simulations, a large number of parameters may be involved, and some parameters may not have point values and may be described by probability density functions
– Model processing: it consists of
– Model analysis (descriptive model analysis (model characterization)) evaluative model analysis (model evaluation)
– Model transformation, and behavior generation

Table 2.7 Types of descriptive model analysis or model characterization

Descriptive model analysis (Model characterization) for
Model comprehensibility
Model documentation
Static model documentation
Dynamic model documentation
Model ventilation (to examine its assumptions, deficiencies, limitations, etc.)
Model usability
Model referability
Model-based management
Model integrity
Model composability

Table 2.8 Types of model evaluations

Evaluative model analysis (Model evaluation) with respect to
Modeling formalisms
Consistency of model representation
Static structure of
Component models
Total system (coupled model, model of system of systems)
Dynamic structure
State transitions, output function(s)
Structural change
Dynamic coupling
Model robustness
Another model (model comparison)
Structural model comparison
Model verification (comparison of a computerized model and corresponding conceptual model)
Checking
Model homomorphism, model isomorphism
Model equivalencing for:
– Any two models
– A simplified and original model
– An elaborated and original model
Behavioral model comparison (comparison of behaviors of several models within a given scenario)
Real system
Model qualification
Model realism (model veracity, model verisimilitude)
Adequacy of model structure:
– Static structure (relevant variables, interface of models)
– Dynamic structure
Adequacy of model constants and parameters
– Model identification, model fitting, model calibration
Model correctness analysis
Dimensional analysis
Model validity
Goal of the study
Model relevance
Domain of intended application(s) (appropriate use of a model)
Range of applicability of a model
Acceptability of a model with respect to its technical system specification

Table 2.9 Types of model transformations

Types of model transformation
Model copying
Model reduction
Model pruning
Model simplification
Structural model simplification
Behavioral model simplification
Model elaboration
Model isomorphism
Model homomorphism
Model endomorphism

Table 2.10 Types of model behavior

Types of model behavior
Point behavior (as a result of)
Computation
Optimization
Search
Trajectory behavior
Simulators
Simulation
Intermittent simulation
Optimizing simulation
Gaming simulation
Structural behavior
Growth systems
Lindenmayer systems (L-systems)
Mixed trajectory and structural behavior

Table 2.11 Types of model behavior generation

Types of model behavior generation by
Numerical techniques
Deterministic techniques
Stochastic techniques
Non-numerical techniques
By symbolic techniques
By analogical techniques
Mixed numerical and symbolic techniques (multi-paradigm modeling)

Table 2.12 Characteristics of simulation as a knowledge generation activity

Based on **types of knowledge processing:** (Simulation is a *knowledge generation activity*)
 From this perspective, “Simulation is model-based experiential knowledge generation” [2].
 Based on this feature

– Advanced simulation environments can combine modeling, model processing, behavior generation, and other types of knowledge processing (integrated use of M&S with optimization, AI, and software agents)

– Combination of simulation systems with sensors and effectors; and switching from simulation to real system operation or vice versa is possible

– Combination of several types of knowledge processing such as soft computing, cognitive, and emotive computing is possible

Table 2.13 Characteristics of simulation as a knowledge-processing activity

Based on **types of knowledge processing:** (Simulation is a *knowledge-processing activity*)

Advanced simulation environments can

Combine modeling, model processing, behavior generation, and other types of knowledge processing such as

– Integrated use of M&S with optimization, AI, and software agents

Combine simulation systems with sensors and effectors; and switching from simulation to real system operation or vice versa

Allow combination of several types of knowledge processing: soft computing, cognitive, and emotive computing

A pragmatically important type of model processing is generation of model behavior which is done in every simulation study. Tables 2.10 and 2.11 outline, respectively, types of model behavior and types of model behavior generation.

Characteristics of simulation as a knowledge generation activity are outlined in Table 2.12.

Characteristics of simulation as a knowledge-processing activity are outlined in Table 2.13.

From the point of view of philosophy of science, simulation supports and enriches modern scientific thinking as advocated by Bacon [15].

2.2 Data

Umang Kant, Mayank Sing

Data plays a pivotal role in the process of simulation. Data works as the input to the simulation models which in turn produce data. Simulations are often used to solve real-time problems, and hence, we need real-time data to make the simulation a success. Simulation process gives solutions to problems by giving clear insights. The data used for simulation will shape the outcome of the process. Hence, it is of utmost importance to be clear about what type of data is to be used.

The choice and value of data used in simulation can have an impact on the underlying mechanisms that control the behavior of the system. In this chapter, we explore and discuss the types of data and various terms related to data with respect to M&S BOK.

Accessible data

Accessible data refers to the data which can be retrieved, modified, copied, or moved from an organization's repository (hard drives, disks, database, data warehouse, cloud, etc.) provided the party (users) accessing the data has proper authority. The data can be accessed on demand as per needs, and authorized users can perform above-mentioned functions at any location as per convenience. So, data access can be defined as the way by which authorized users can get access to this data and its location in an authenticated manner approved by the organization having that data. This data can either be at rest or in motion. When data is at rest in a repository, we can access the data in two ways: (i) *sequential access*: where seek operation is used till the required data is found and (ii) *random access*: data can be retrieved from anywhere in the disk.

Actual data

Actual data is the data which has been interpolated using the current parameters, hence making it more fluid data. Actual data is different from *historical data*, as well as *forecast data*; where the historical data is the collection the data which has happened in the past and has records of its existence, whereas forecast data is the prediction based on the correlations of the historical and current trends.

Adjusted data

The principle behind the concept of adjusting data is that the data is adjusted to satisfy the constraint which is believed to be true and is a fact. Hence, adjusted data can be defined as data which has had some modifications and manipulations to adjust in the required specifications of the application. Three of the most significant methods of adjusting the data are (i) *weighing data*, (ii) *variable respecification*, and (iii) *scale transformation*.

ALSP data

ALSP stands for aggregate level simulation protocol. This protocol facilitates interoperating the simulations with each other. It manages the adaptable demands of the system with respect to the application in use. ALSP data refers to the data which supports the various simulations and their relationship with each other.

Ambiguous data

Ambiguous data refers to the data having same representation but different meaning or value. Ambiguous data leads to ambiguous information which in turn affects the process of decision making. Ambiguous data can be mined using *multi-instance multi-label learning* and other algorithms.

Auditable data

Auditable data is set of data which is forwarded to the process of data auditing. Data audit is the process to auditing the data to evaluate the utility and quality of the data being used for a specific purpose. The process of data audit requires considering key metrics to assess the quality of the data set. The main objective to carry out this process is to check if the data set fits the organization's data for a particular purpose, i.e., to check how the data set affects the performance, cost, and profits/loss of the organization.

Bivariate data

Bivariate data refers to the data of two related variables. For example, sale of ice cream and temperature are related; similarly, the booking of hotel rooms is dependent upon the weather and vacation months. Such examples refer to bivariate data. Bivariate data is different from univariate data which is dependent on only one variable.

Calibrated data

Calibration refers to the process of comparing a device under test (DUT) on an unknown value with respect to a standard of a known value. Calibrated data can be defined as the data upon which the calibrated settings have been applied. Calibration settings are flexible, and they are applied after applying the suitable data settings.

Certified data

Data is interpreted as certified data when a certified data analyst (CDA) acquires, cleanses, processes, and analyzes the data. Certified data helps the certified data analyst in making efficient decisions and producing business reports in various industries such as retail, telecommunication, financial, medicine, tourism, and others.

Coarse data

Coarse data can be interpreted as data that one observes not the true value of variable of interest but rather a subset of the sample space in which the true data lies. When the data is not to be found, their true values are known to lie somewhere in their sample space. The process of coarsening is not stochastic in nature. The degree of coarsening is pre-decided and is known in advance.

Complex data

Complex data is a type of data which indicates the level of difficulty while translating it into some business value. As the name suggests, complex data is difficult to collect, model, and analyze than simple data. Such data requires various tools for interpretation and processing. Data falls into the category of complex data when it is 'big' in nature and when it is coming from various disparate sources.

Consistent data

By consistent data, it is understood that the data is produced in a regulated and predictable manner. Consistent data is constant over time and relationship with other elements of the system environment. Each time the data is run on a particular system or application, it produces the same result, i.e., there is no data discrepancy. Data is considered to be consistent if it is formatted in a manner. A system is said to include consistent data if it does not contain contradiction or does not allow the users to infer any contradiction with respect to the data.

Correct data

As the name suggests, correct data is exact in nature, i.e., it is accurate and is free of faults and inconsistencies with respect to the system or the application, where that data is to be used and processed.

Current data

Current data lets the organization team to view the state-of-the-art analytics data even before the data is completely processed and confirmed for decision making. It is in the nature of current data to display the metrics in least amount of time, which in turn provides actionable data which is further used for decision making. In reports, the current data is enabled by default to support quick decision making.

Digital data

Information which can be stored on the machine in the form of collections of 0s and 1s, i.e., in binary language is called digital data. A sequence or collection of 0s and 1s is forwarded to digital machines because these machines do not understand human understandable language (high-level language); hence, it is must to convert the information into low-level language and then covert the low-level language using a compiler into a sequence of 0s and 1s. Digital data is different from analog data.

DIS protocol data

DIS stands for distributed interactive simulation. DIS refers to a protocol that has been specifically designed to facilitate communication between heterogenous simulators built by different manufacturers. The protocol data unit (PDU) is the DIS format for truth conveyance. It is an ambiguous data format for communicating a particular event or specific piece of information.

Environmental data

Environmental data refers to that data which is solely based on the (i) measurements of environmental characteristics such as pressure and temperature, (ii) state of the environment, and (iii) the environment impacts on the ecosystems. Environmental statistics are also often considered as environmental data. While understanding environmental data, we need to understand what data we need to collect and analyze and then break down the types of data. The environmental data can be

classified into following types: (i) continuous, (ii) count (simple or categorical), (iii) proportion, and (iv) binary or unary, etc.

Evaluation data

Evaluation data refers to the compilation of data and the use of consistent analysis methods. Evaluation data includes identification of research goal, existence of research problem, methodology type, data compilation processes, sample data, and data analysis techniques and outcomes. If we are trying to find out: *how many? how much? what percentage? how often? what is the average amount?*, etc., then we must choose quantitative methods. If we are trying to find out: *what worked best? what did not work well? what do the numbers mean? how was the project useful? what factors influenced success or failure?*, etc., we must choose qualitative methods.

Exchange data

Exchange data refers to the data being shared among different stakeholders or different computer programs. Here the data is structured under a source schema and transformed into a target schema. It provides data points from various parts of the world to support data marketing and advertising. Here we can use previously unavailable or outdated data and use it to power the marketing campaigns.

Experimental data

In science and engineering, experimental data is the data produced by a test method, a measurement, an experiment design, or quasi-experimental design.

Haptic data

Haptic data is the data being generated by haptic technology. The aim of haptic technology is to simulate the sensory environment and enable the users to touch, feel, and manipulate virtual objects in the environment through haptic interfaces as true to the reality as possible. By incorporating perceptual cues such as feel, shape, texture, stiffness, and friction, one can depict properties of virtual object.

Hardwired data

By hardwired data, we mean the data which is not flexible and will not thrive in different environments. Hardwired data comes wired up by the developer or by the system before it can be used, and during the use, there is no possibility of changing its properties.

Heterogenous data

Data which is generated by Internet of Things (IoT) is heterogenous data. These types of data come with high variability of data formats and data types. These data are ambiguous in nature, and since they are collected over the Internet with sensors, i.e., they are collected on a large scale, there is no guarantee of their quality. This type of data needs to be analyzed before use. Heterogenous data falls into the

following categories: (i) *syntactic heterogenous data*, (ii) *conceptual heterogenous data*, (iii) *terminological heterogenous data*, and (iv) *semiotic heterogenous data*.

Historical data

In broad context, historical data can be defined as the data which is collected from past events and situations pertaining to a particular application. This data is generated either manually or routinely within an enterprise.

HLA protocol data

HLA stands for high-level architecture, and HLA protocol data refers to more recent standard for interoperability among simulations. This data refers to architecture with a set of API standards as opposed to a networking protocol like DIS. Real-time infrastructure supports and implements the HLA protocol data and transports data from one federate to another.

Input data

Input data serves as input to the system, device, or programs.

Instance data

Instance data completely describes the state of object in question. It consists of all the internal states of the object so that it almost becomes synonymous to the object itself.

Instance meta-data

Instance meta-data discusses the instance. This data can be used to manage the current running instance. Instance meta-data can be divided into categories depending upon the application in question such as events, partners names, and security levels. Instance data can also be used to access user data specified at the launch of the instance.

Irrelevant data

Irrelevant data refers to the data which is not connected or relevant to the application being dealt. Using irrelevant data will lead to change in the desired outcome. Irrelevant data can also be understood as unimportant data.

Legacy data

Legacy data refers to such data which comes in the category of enterprise essential information. This data is stored in some old format in computer system. Legacy data is usually used in government schemes or in industries. This data is generally available online depending upon the use. It is important to handle the legacy data by following some steps: (i) *develop data error handling strategy*, (ii) support read-only legacy data access, (iii) encapsulate legacy data access, etc.

Livewired data

Livewired data is complete opposite if hardwired data. As opposed to hardwired, livewired data provides huge flexibility and possibility of changes as per the environment. This type of data can thrive in any type of environment as it learns to adapt to the environment, and hence, the properties are also not fixed.

Meta-data

Meta-data refers to data about data.

Model data

Model data is the data used in a data model, where model data and other related elements are organized together. The related data follows model's standardization.

Monitoring data

Monitoring data is obtained by the method of evaluating and reviewing the data at every step. The quality of the data is to be ensured and monitored so that it is fit for the purpose. For the monitoring purpose, we can use data monitoring software which helps in tracking the data for the use.

Multisample data

Multisample data is obtained in those instances where enough experiments are performed so that the reliability of the results can be assured by statistics.

Noisy data

Noisy data can be referred to as corrupt data or meaning less data. This data cannot be understood and interpreted by the machines used in the organizations. Other issues with noisy data are that it wastes the storage capacity of the machine and yields incorrect results. Noisy data is generated by human or computer error, faulty instruments, data transmission errors, naming convention inconsistencies, inconsistent formats, etc.

Non-stationary data

The data which is unpredictable and hence cannot be modeled or forecasted is known as non-stationary data. They yield spurious results and hence indicate that they are fake or nonexistent in nature. They must be converted into stationary data in order to obtain better results.

Notional data

Notional data refers to such data which exists only in theory, i.e., it does not exist. It is such data which has not been scientifically proven but still can be used for experimentation purposes.

Observational data

Observational data is collected by systematic methods and hence is valuable from the point of view of a particular research. Observational data, as name indicates, is collected by observation. There are many observations which yield observational data: (i) *participant and non-participant observation*, (ii) *simple and behavioral observation*, (iii) *direct and indirect observation*, and (iv) *covert and overt observation*.

Obsolete data

Information which is either outdated or no longer in use, incorrect, or incomplete is termed as obsolete data. This type of data is usually replaced by new more accurate data.

Original data

Data or information relating to original material is referred to as original data.

Output data

Output data refers to the output value of the system or software. This data reports the extent to which the intervention impacts.

Perceived data

Perceived data measures the report of competency attainment, beliefs, and attitudes along with perceived gains in knowledge.

Persistent data

It refers to the data which does not change with time, does not change across systems, and memory. This data is non-volatile and durable even with the change in software and devices, and it exists from one instance to another. This data is opposite of transient or dynamic data. This type of data is also called as dimensional data with respect to data warehousing.

Qualitative data

Qualitative data describes the quality of the object. It characterizes and approximates the object. It is non-numerical in nature and is collected through methods of one-on-one interviews, observations, record keeping, focus groups, case studies, etc., and qualitative data can also be termed as categorical data with respect to statistics.

Quantitative data

Quantitative data represents the numerical value of the object. This data can be measured with respect to numbers or counts. This data is also termed as numeric data. Quantitative data can be further classified into (i) *digital data* and (ii) *continuous data*. This type of data can be used in census, data projection, data prediction, etc.

Real-world data

Real-world data refers to the data which has been collected through observation and not through experimentations. Real-world data plays a pivotal role in real-time applications like healthcare systems, medical and clinical research, weather prediction, air traffic control systems, etc.

Reference data

Reference data can be defined as the data which defines structures of other data stored in the database. They are used to categorize or classify other data. They are usually static in nature or are rarely or slowly changed over period of time.

Relevant data

The data which is useful and not distracting is termed as relevant data. They are used at the runtime to complete the work in hand. Relevant data is indisputable and can be used to create strong strategies. Relevant data is useful to optimize and quantify the research.

Semantic data

Semantic data is useful in semantic data model which is a method of structuring the data in order to represent it in logical way. Semantic data adds meaning to the data and the relationship among other data in the model.

Sensor data

Sensor data can be defined by the data which is the output of a sensor (things in IoT) that identifies and acts on the input from the environment. This data is collected through sensors which in turn are connected through gateways, and collected data is transferred to the cloud or fog.

Sensory data

Sensory data is the data which is made available to the person through its sensory organs. In case of machines, the sensors are used to sense and collect the data. Sensory data can also be referred to as the physical effects of the external environment on the senses.

Significant data

Significant data can be interpreted as relevant data. The data which is useful and not distracting is termed as significant or important data. They are used at the runtime to complete the work in hand. Significant data is indisputable and can be used to create strong strategies. It is useful to optimize and quantify the research.

Simulated data

The data produced by data simulation process is called as simulated data.

Single sample data

Single sample data is the type of data in which few uncertainties may not be discovered by repetition. They are the opposite of multi-sample data.

Smooth data

Data obtained by the process data smoothing is called as smooth data. Smooth data is free from outliers and noise. Algorithms can be used to perform data smoothing process.

Source data

Source data is the primary location from where the data comes. Also termed as data source, it can be data set, database, and Excel spreadsheet.

Stationary data

Stationary data is predictable in nature, and hence as opposed to non-stationary data, they are predictable in nature and can be forecasted.

Technical data

Technical data is a recorded information of a scientific or technical nature. It can be defined as recorded information regardless of the ways of recording. Technical data can be represented in a technical data package.

Test data

Test data is the data which is given as input to a software during test execution. It is used for testing purpose. It is the data which either gets affected by the software or affects the software.

Theoretical data

Theoretical data is collected by the process of theoretical sampling for generating theory. They are abstract in nature.

Time indexed data

Data which is naturally evenly spaced in time is called time indexed data.

Traceability data

Traceability data is the information which refers to completeness of information about every step of the process.

Transient data

Transient data is in contrast to persistent data. It is volatile in nature as it is created within an application session. As soon as the application session is over, this data is discarded or reset to their default state.

Additional definitions and explanations on these topics are published in [16–23].

2.3 Models and Modeling Formalisms

Paul Fishwick, Mamadou Kaba Traoré

We begin with a broad cultural introduction to the idea of modeling and follow this with a description of how models can be formalized.

A model is a simplified representation of something in the world. A toy car represents an actual, real-sized automobile, but the toy car leaves out much detail while providing relatively inexpensive and flexible usability (i.e., handling the car). We say that the toy car *models the automobile*. Models are studied in every field, and there are significant differences depending on these fields; however, all models capture a simplified representation of something else. This overarching goal is the same regardless of discipline. Within single disciplines, there is much literature. For economic models, for instance, [24] presents a comprehensive review.

Model taxonomy has to be constructed from the ground up, through induction of any use of the word ‘model’ within the broad literature. Such a taxonomy will not be explicitly defined here, but we introduce the sorts of models available. Modeling is fundamentally cultural. This cultural aspect of modeling is partly due to discipline. The electrical engineer may model an analog or digital circuit at different levels of abstraction. The languages used by the engineer have a cultural foundation. Engineering as a discipline is strongly visual—in the form of schematics and diagrams. It comes as no surprise, then, that when the engineer thinks of modeling, they frequently refer to the dynamics of an object through liberal use of diagrams and drawing and contrast this culture with the culture of the mathematician. The mathematician and the physicist, who relies on applied mathematics, tend toward canonical modern mathematical notation, which is lexical. This is not to suggest that the mathematician does not use diagrams, but their work is most naturally found in text-based notation. These notations have evolved over the centuries and will continue to evolve. The ways of thinking, and so ways of modeling, differ among cultures.

The following represents an incomplete list of types of model by topic area, while referring to specific examples:

- (1) The scale model—this is perhaps the most widely used type of model. The model is a scaled-down, simplified, geometrically similar, object when compared to the object being modeled. We say that the scale model *looks like* the actual object; the representation is similar from a visual perspective. An example is the town model of Bourton on the Water [25]. While the Bourton village captures the *model of* category [26], other model villages [27] are examples of *model for*.
- (2) The art model—within the arts, a model is either a quick sketch or prototype of the target object or a human who models one example of an ideal human (e.g., the model who sits and poses for artists who sketch and paint). Examples are found in Bernini’s terra-cotta models [28] with a sample exhibition [29].
- (3) The engineering model—based on mathematical notation or the use of diagrams [30].

- (4) The mathematical model—based on mathematical notation indicating the modeling of a real scenario [31].
- (5) The logic model—based on an interpretation of a formal language satisfying specified axioms (e.g., a mapping or set of equivalences that cause one or more logical sentences to be true) [32].

The case of system dynamics (SD), initiated by Forrester [33], captures the cultural essence of modeling. In SD, there are phases in the modeling process. While all models can be said to begin with abstract concepts which eventually lead to more detailed concepts and formalisms, SD explicitly defines these transitional steps, whereas most modeling languages do not. SD begins with natural language where concepts are drawn on a board. These concepts are then connected with arrows. The arrows are labeled with plus and minus signs to define cause/effect or increase/decrease. Then, the directed graph of arrows creates negative and positive feedback loops. The process so far yields a causal loop diagram. The diagram must be annotated so that specific nodes in the graph become levels or rates, or one of a small set of other possibilities. This is then converted into a flow graph. Finally, the flow graph is translated into a set of ordinary differential equations. These equations are then translated into computer language and can be executed on the computer. Not only does SD capture an evolutionary process for a certain type of modeling, yielding ordinary differential equations, but each level represents a different cultural lens for the target objects being modeled. The starting point of natural language provides a common familiarity since anyone who knows English, for example, can create such a model. The diagrams have an engineering cultural appeal. The equations represent the formal foundation and for the casual user may appear daunting. Incremental modeling, or model engineering, involves baby steps at first, gradually moving toward an ever-refined mathematical model.

Models are social and cultural in addition to having different forms of representation. These representations reflect the cultural differences of those who model. Some models are simple, others complex. Some models use two-dimensional or three-dimensional materials. We also need to be aware of *model for* versus *model of*. A model can be made from a preexisting object (i.e., the target of modeling) or the model can be a prototype for something not yet constructed.

While models, across all disciplines, are deeply cultural with different types of representation and detail, there are notable standards based on mathematical notation used frequently in science and engineering. We refer to these models as formal models, where the formality derives from mathematical notation as a *lingua franca* for many disciplines.

Formalisms provide the means to explicitly express models. While browsing the whole space of all formalisms seems not realistic, there has been attempts to classify them along criteria that can help choosing the right ones in a given modeling situation. One such approach is proposed under the umbrella of multimodeling [34], where the following classification of abstractions is done, and for each class of abstractions, a family of adequate formalisms is indicated:

- Conceptual models describe, at the highest level of abstraction, the general knowledge about the system under study. Their purpose is to highlight system entities (or classes), as well as the relationships between them. They, therefore, constitute a knowledge base for subsequent abstractions. UML and natural language are examples of adequate formalism for this abstraction level.
- Declarative models provide systems description in terms of sequences of events that move the system from one state to another. This level of abstraction is adequate to represent the dynamics of objects identified in the conceptual model. Sequential automata (for passive objects) or Petri nets (for active objects) are adequate formalisms for this abstraction level.
- Functional models provide system descriptions in terms of sequences of functions (the outputs of the ones feeding the inputs of the others), arranged such that the final treatment performed by the system is found downstream of this sequence. Such a level of abstraction specifies the flow of processing within the system. Queueing networks or functional block diagrams are examples of adequate formalism for this abstraction level.
- Constraint (or conservative) models provide system descriptions in terms of laws and constraints. Differential equations or difference equations are examples of adequate formalism for this abstraction level.
- Spatial models focus on describing the decision rules for systems operation. The main idea, at this level of abstraction, is to represent global decisions as the fruit of the interactions of multiple local decisions. Cellular automata or multi-agent systems are examples of adequate formalism for this abstraction level.

A more system-theoretic approach to formalism classification emphasizes on the concepts of time and state [35] and distinguishes the following cases:

- The evolution of the state of the system of interest is specified as a continuous process in a continuous time base. Formalisms that allow such a specification fall under the label DESS (for “Differential Equation System Specification”).
- The state dynamics is specified as a continuous process in a discrete time base. Formalisms for such a specification fall under the label DTSS (for “Discrete Time System Specification”).
- The state evolution is specified as a discrete process. Therefore, the time base is necessarily discrete even if it can take values in a continuous (respectively, discrete) set, i.e., the set of reals or a subset (respectively, the set of integers or a subset).

Modeling formalisms are strongly related to the modeling objectives. The latter are concerned with the overall process that makes use of models, while the former are tools used among others within this process to describe models. Therefore, the use of whether a unique formalism or multiple formalisms is guided by the modeling objectives, including the questions to be answered about the system, the properties of the system to be analyzed, the capabilities of available model solver, the analyst's experience, etc. When multiple levels of abstractions are at stake, this

may lead to the use of various formalisms, each of them most suitable to a level of abstraction. Such heterogeneity raises issues related to semantics alignment of the overall model, like consistency between the different specifications, overlapping descriptions, or unifying semantics. At the other side, a single formalism, thought free from semantics alignment issue by construction, is rarely appropriate alone for all levels of abstraction.

In order to conciliate the semantics unity of a homogeneous situation with the expressive power of heterogeneous specification, unifying modeling approaches have been defined, ranging from formalism interfacing (i.e., the use of a third party as a glue to combine various formalisms) as proposed in multimodeling [34], to formalism federation (i.e., the use of complementary formalisms to cover various viewpoints of the same system) as proposed by UML [36], formalism subsumption (i.e., the act of defining a model of computation that subsumes the semantics of different formalisms) as proposed by DEVS [37], and formalism weaving (i.e., the creation of a pivotal formalism by weaving the metamodels of existing formalisms to create a new domain-specific language) as proposed by HiLLS [38].

2.4 Model Engineering

Lin Zhang, Yuanjun Laili

Model engineering (ME) or named as model lifecycle engineering (MLE) deals with the model lifecycle credibility and scalability problem for complex systems, especially systems of systems. ME is supposed to be taken as a sub-discipline of M&S, which aims to provide standardized, systematic, and quantifiable management and control to guarantee the credibility of the model lifecycle. Moreover, ME can be used not only in the domain of M&S, but also other fields that need modeling and model management.

2.4.1 Model Life Cycle

The life cycle of a model is shown in Fig. 2.1. It contains six phases, i.e., problem definition, model design, model construction, VV&A (verification, validation and accreditation), model implementation, model evolution and reconfiguration, and model maintenance [39].

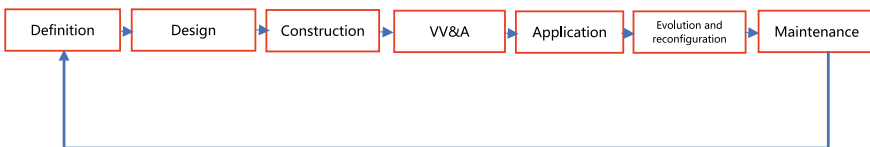


Fig. 2.1 Lifecycle of a model

Problem definition is to define the specification of requirements. Model design is to design the framework and relationships of different parts of the system. Model construction is to build the model using description languages or mathematical formalisms. VV&A is the process of verification, validation, and accreditation. Application is to apply the model in a specific simulation environment. Evolution is an incremental adaptation process to make the specific model more scalable and credible to the current system requirements, including modification and refinement of model parameters and relationships. Reconfiguration is to change parts of the model during its runtime. Maintenance is to manage data, parameters, and different versions related to all other phases of the lifecycle.

2.4.2 Definition of Model Engineering Model Life Cycle

Based on this cycle, model engineering is defined as a general term for theories, methods, technologies, standards, and tools relevant to a systematic, standardized, and quantifiable engineering methodology that guarantees the credibility of the full lifecycle of a model with the minimum cost [40, 41].

2.4.2.1 Model engineering regards the full lifecycle of a model as its object of study, which studies and establishes a complete technology system at the methodology level based in order to guide and support the full model lifecycle process such as model construction, model management, and model use for complex systems.

2.4.2.2 Model engineering aims to ensure credibility of the full model lifecycle, integrate different theories and methods of models, study and find the basic rules independent of specific fields in the model lifecycle, establish systematic theories, methods, and technical systems, and develop corresponding standards and tools.

2.4.2.3 Model engineering manages the data, knowledge, activities, processes, and organizations/people involved in the full lifecycle of a model and takes into account time period, cost, and other metrics of development and maintenance of a model.

2.4.2.4 Here the credibility of a model includes functional and non-functional components. Functional components are measurement of the correctness of functions of the model compared to the object being modeled. Non-functional components include features related to the quality of a model, such as availability, usability, reliability, accuracy, integrity, maturity, ability of modelers as well as management of modeling process. Credibility is a relative index with respect to the purpose of modeling and simulation. Evaluation of credibility includes objective and subjective evaluation. Objective evaluation is mainly based on data and documents, while subjective evaluation is mainly based on expertise. Quantitative definition and measurement of credibility will be one of the most important research topics of model engineering.

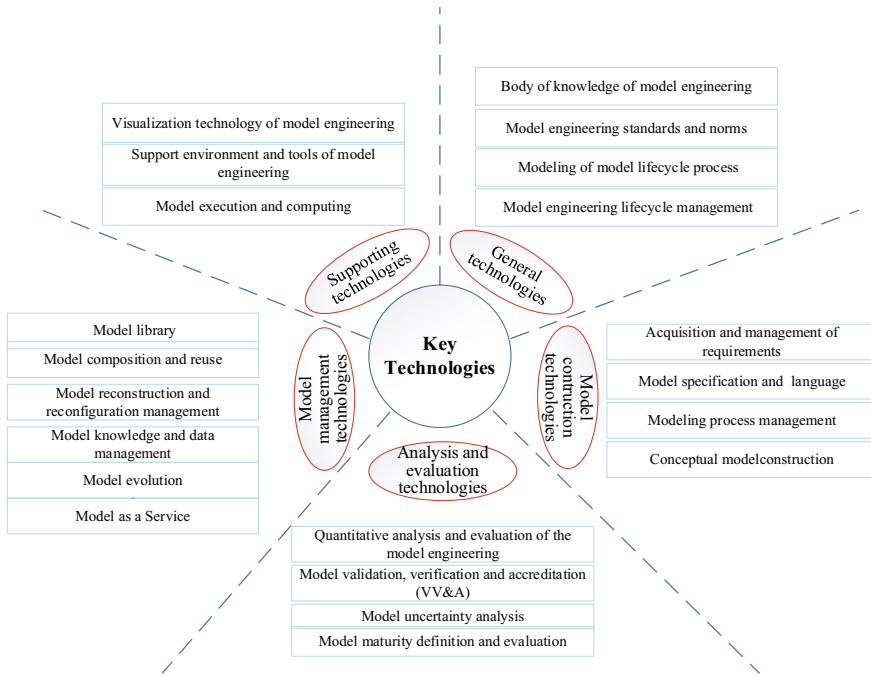


Fig. 2.2 Key technologies of model engineering

2.4.3 Key Technologies of Model Engineering

According to the framework of the body of knowledge of model engineering given in (Zeigler and Zhang 2015; [41]), technologies involved in ME can be divided into the following categories (Fig. 2.6) including general technologies, model construction technologies, model management technologies, model analysis, and evaluation technologies, supporting technologies. Some key technologies in the categories will be discussed in this section (Fig. 2.2).

2.4.4 General Technologies

2.4.4.1 Modeling of Model Lifecycle Process

In accordance with the standards of ME, modeling of model lifecycle process means to build a structural framework of activities that usually happen in the lifecycle. The framework is a visible pipeline to show the state of a model related to the key stages, key elements, and key data of its lifecycle management. It is also designed as a reference to evaluate the lifecycle cost and comprehensive efficiency and improve both the model and the management strategy.

2.4.4.2 Model Engineering Lifecycle Management

The lifecycle management of model engineering is carried out for managing data (development data and runtime data), knowledge (common knowledge shared by different models and the domain knowledge), activities, tools (especially M&S tools and model evaluation tools), and person (the modeler, the tester, and the user). Data/knowledge management technology focuses mainly on the data and knowledge in model, runtime environment, and the whole model lifecycle. It includes the methods for key data extraction during the model engineering lifecycle, knowledge classification from multi-disciplines, information learning throughout modeling and simulation, and data/knowledge storage for further improvement.

In the near future, we expect that the number of multi-disciplinary models will grow, and the assembly and disassembly of a systems, data, and models will continue to become more complex.

Accordingly, data mining strategies and knowledge extraction algorithms used in ME must become much more: (1) scalable, to adapt to a wider arrange of domain information, (2) efficient, to implement intelligent system construction, and (3) stable, to ensure credible simulation and model management.

ME lifecycle management also consists of monitoring the processes of model reconfiguration, evolution and maintenance, and the multilayer optimization of modeling practices, operational workflows and maintenance schemes to realize an efficient risk/cost control and speedup throughout the whole lifecycle of a model.

2.4.5 Model Construction Technologies

A large amount of research on model construction (modeling) has accumulated over the years in the M&S domain. From the point of view of model engineering, some issues for modeling methods are of most concern. Such issues include (1) acquisition and management of model requirements, (2) model specifications and modeling languages, (3) modeling process management, and (4) conceptual model construction.

2.4.5.1 Acquisition and Management of Model Requirements

The model lifecycle starts with requirements. Accurate requirement acquisition is the key to credible M&S. However, requirement acquisition and management are very challenging due to uncertainty and ambiguity in the systems being modeled. Research on requirement acquisition is needed to improve the means to extract, describe, parse, and validate requirements via automated or semi-automated means. Similarly, research is needed on management of requirements to formulate how to reflect changing requirements to influence model construction and maintenance in an accurate and timely manner.

To acquire accurate model requirements for system simulation, we need to get as much information as we can to understand (a) the underlying modeling objectives of the simulation, (b) the nature of the targeted system, and (b) the kind of environmental conditions that are required. Therefore, an analytical strategy is

particularly important in which we hierarchically decompose the structural requirements for further design and model matching.

Additionally, system features (user demand, system structure, and environmental conditions) extracted from the above strategy must be stored and managed by category of the facet. When a new requirement comes in, these facets will be used to match similar models and existing domain knowledge to support rapid system construction.

2.4.5.2 Model Specification and Language

Model specification is informed by the detailed description of system simulation requirements, model input/output, model functionalities, model activities/states and the related domain rules. The model requirement is not only a documental or structural description about what kind of model we need, but also a simple and uniform representation of the general features possessed by a targeted system. These features should specify some common rules corresponding to system components and their interconnections. Similarly, at the component level, the specification for the meta-model (which performs some domain-independent functionalities/states/activities) and the domain rule (which represents some domain-dependent state transformation mechanisms) should be prebuilt. By dividing a general domain model into meta-model and domain rule, a domain model is easily assembled and disassembled for efficient reuse. In addition, this division enables the modeling data/knowledge to be clearer and easier to manage.

Although a model can be built by different sorts of lower-layer model languages, a unified upper-layer language is still necessary for engineers to efficiently construct a new system model or transform an existing one to meet the simulation demand. Following the vision of the above specifications, an upper-layer model language should also clearly describe the outer structure and inner behavior of a system and be able to transform into multiple lower-layer model languages.

2.4.5.3 Modeling of Process Management

Two kinds of efforts are necessary to guarantee the credibility of a model. One is to do VV&A after the model is built. The other is to manage and optimize the modeling process. VV&A has important implications to discover model problems and defects, but it clearly does not and cannot solve the problem of how to acquire a correct model. Especially for complex systems, due to the complexity and uncertainty of the system, the modeling process can be very complicated, which makes VV&A of a model also extremely difficult. Even if the defects are found via VV&A, the modification of the model will be very difficult and expensive. Therefore, it is very important to structure and optimize the modeling process. Consequently, methods are needed to measure the degree of formality and optimization (maturity) of modeling and simulation processes. A highly structured level of organizational capabilities and the use of proven processes applied to modeling can guarantee, to a large extent, the credibility of a model. (Fujimoto et al. 2017).

Capability maturity model (CMM) and CMM integration (CMMI), originating in software engineering, can be introduced to establish a capability maturity model

for the modeling and simulation process (M&S-CMMI) (Zhang and McKenzie 2017). Such an approach can enhance both the management efficiency and personnel capabilities in high-quality model construction and management. Related research opportunities include M&S-CMMI evaluation, optimization, risk analysis and control of modeling processes, notional mappings with CMMI, etc.

2.4.6 Model Management Technologies

Model management consists of core methodologies and technologies that guarantee highly efficient and credible composition, sharing, reuse, evolution, and maintenance.

2.4.6.1 Model Library

A scalable model library is key to implement efficient model engineering. It should be able to handle heterogeneous models by using a formal description language, recognize multi-disciplinary model features, and enable a fast indexing and location of similar/suitable model for a task specification. The main techniques with respect to model library are model classification criteria, model storage mode, model indexing schemes, and ways of searching for models. In contrast to a database, the model library stores and processes not only model descriptions, but also their instances and interconnection relationships. Currently, there is lack of techniques for establishing a model library.

2.4.6.2 Model Evolution

Model evolution is one of the most important innovations proposed in ME. From the separated meta-model and domain knowledge point of view, ME aims at using models as a service and enabling them to autonomously evolve to new improved version instead of through manual refinement. Borrowing the idea of incremental learning, model evolution refers to an incremental adaptation process to make the specific model more scalable and credible to the current system requirements.

The factors including parameters, states, behaviors, and functions in meta-model and different sorts of domain knowledge such like additional domain parameters, action rules, constraints, and domain-related functions are all able to be updated as modules of a simulation system, as shown in Fig. 2.3. To enable the meta-model and domain knowledge to update autonomously over time, we need to establish dynamic connections between system requirements and these lower-layer factors in line with the historical data from the model lifecycle. These connections, which can be trained by the existing incremental learning algorithms and intelligent multi-agent system-based strategies, will then guide the factors of a model to change toward the ones connected to the most similar requirements.

2.4.6.3 Model Reconfiguration

Different with model refinement in the stages of application and maintenance, model reconfiguration means to change part of model during its runtime. It is an

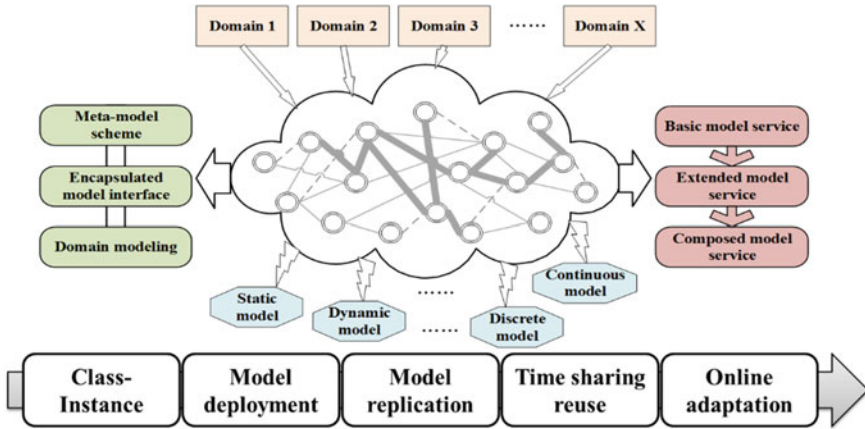


Fig. 2.3 Cloud-based servitization of simulation models

important basis to implement model reuse with a minimum rollback design cost. According to the above-mentioned model specification, a model should be designed to hold multiple functionalities and flexible interfaces. How to dynamically choose suitable functionalities and domain knowledge to ensure accurate response in simulation is the main concern of model reconfiguration methods. Specifically, it can be divided into lower-layer reconfiguration and upper-layer reconfiguration.

Lower-Layer Reconfiguration

The lower-layer model reconfiguration method is only for a meta-model which is independent with domain knowledge. It includes functional reconfiguration, structural reconfiguration, and parameter reconfiguration, as shown in Fig. 14. Specifically, structural reconfiguration refers to combined multiple meta-model to form a larger one with more functionalities.

Upper-Layer Reconfiguration

On the contrary, upper-layer reconfiguration is directly related to domain knowledge and the practical simulation environment. Thus, it can be divided into domain-related reconfiguration and simulation-related reconfiguration. The domain-related part is responsible for selecting add-on domain knowledge (i.e., domain functionalities, domain parameters, and constraints) to a model, while the simulation-related reconfiguration is set up to determine environmental parameters and simulation engine-related settings to assure a correct and fluent simulation process. Obviously, model reconfiguration is a complex dynamic optimization problem, in which the two-level variables can be either determined in two steps or in one time.

2.4.6.4 Model as a Service

With the development of cloud-based technologies, a heterogeneous model with its execution engine can be integrally encapsulated as a service. That is to say, not only

a meta-model and a domain component model can be encapsulated, a composed system model is also capable of being encapsulated. This makes the execution of different sorts of model easier in any cloud-based environment. To implement flexible model sharing, the scheme of cloud-based servitization of simulation model is illustrated in Fig. 2.3.

(Servitization is a new concept that has two meanings from IT and business perspectives. From the IT perspective, it means the service encapsulation of objects with service-oriented technology. From the business perspective, it is defined as “the innovation of organization’s capabilities and processes to better create mutual value through a shift from selling product to selling Product-Service Systems,” where a product-service system is “an integrated product and service offering that delivers value in use” and a “servitized organization is one which designs, builds and delivers an integrated product and service offering that delivers value in use” (<http://andyneely.blogspot.com/2013/11/what-is-servitization.html>).

First of all, a uniform servitization template (i.e., a service class) should be designed previously for different levels of model. When a system construction requirement arrives, the suitable models will be deployed or replicated into different virtualized cloud resources to support a distributed simulation.

2.4.6.5 Model Composition

Model composition is a technology established upon the flexible model reuse scheme. It is designed to realize more intelligent model collaboration and system construction when the number of models is too large to be implemented with manual selection. In the research of model composition, two critical problems are how to match suitable models to form a valid candidate set and how to select the best models for a collaborative system simulation. The former matching problem can be solved by some feature-based or domain-based model classification and model clustering methods, while the latter model selection has to be considered in different conditions, i.e., offline condition and online condition.

Offline Model Composition

Different with the general service composition, the collaboration between models are not usually perform in a strict sequential or directed acyclic manner. Feedback cycles and concurrencies may exist simultaneously in their collaborative topology. Therefore, traditional algorithms designed for service composition may not applicable. An offline model composition method should be able to recognize each kind of connection in a specific collaborative topology (extracted from a standardized system simulation requirement) and generate feasible solutions with the consideration of sequencing rules, cycling rules, and concurrency rules between the candidate models.

Online Model Composition

Online model composition is executed during the real-time simulation process based on a given offline composition scheme. The main workflow of an online model composition method is drawn as shown in Fig. 2.4. Specifically, it is driven

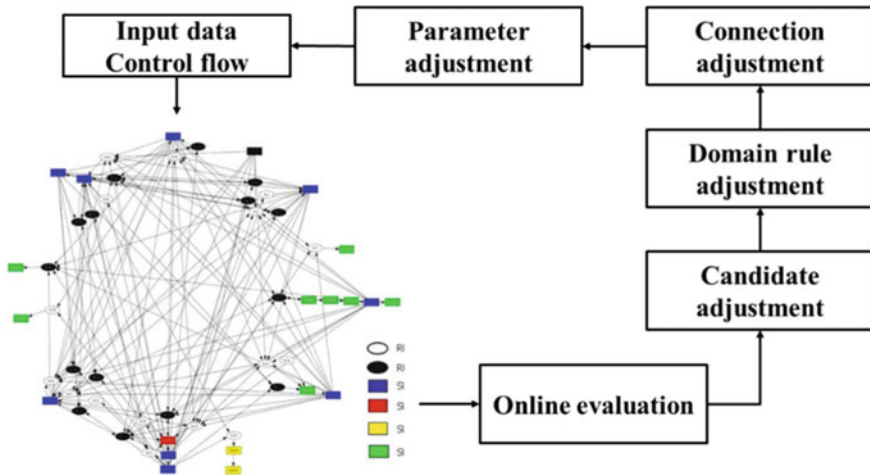


Fig. 2.4 General workflow of an online model composition method

by online evaluation of the current system state compared with a desired state. If online adjustment threshold is reached, the method will perform the adjustment in four steps, i.e., candidate adjustment, domain rule adjustment, connection adjustment, and parameter adjustment from the top down. After the online refinement, the evaluation model will continue to monitor the system state and determine whether a further modification is required. In other words, an online model composition method should be performed at a high speed and provide a feasible solution at different levels and thus is more difficult to design.

2.4.7 Analysis and Evaluation Technologies

Model evaluation is a very traditional topic in the domain of M&S. In ME, it means not only the VV&A of a model, but also the evaluation of the whole process of ME.

2.4.7.1 The VV&A of a Model

In the past decades, different authoritative organizations and researchers have established a few standards for the VV&A of a non-separable model. However, little research has been done on the evaluation of models in a composed situation and in its further maintenance process. As demonstrated in Fig. 2.5, the bridge between the existing model evaluation indicators and the models in different layers of system construction is a key to implement the efficient evaluation of model lifecycle.

In addition, most current research focuses on qualitative analysis and quantitative and formalized analysis methods are lacking, so VV&A quantitative analysis and formalized analysis technology are still main research content in the model engineering.

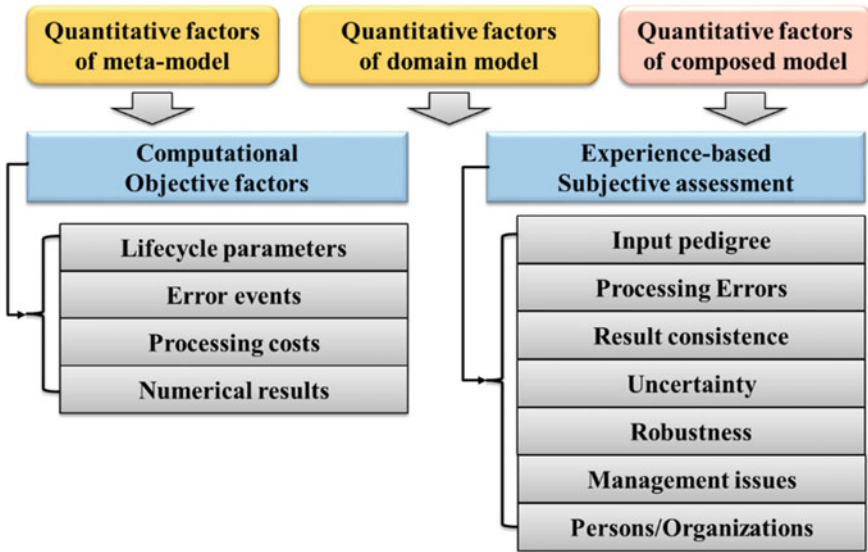


Fig. 2.5 Gap exists between the existing evaluation indicators and the models in different layers

2.4.7.2 The Evaluation of the Whole Process of ME

The quality of the ME process will directly determine the quality of a model in its design, implementation, application, and maintenance. To ensure the control and management quality of ME, key issues from both the control stages and the management process should be extracted first for the construction of evaluation indicators. With a suitable set of indicators to fully cover the whole process of ME, existing expert scoring mechanisms such like fuzzy AHP and TOPSIS can be directly applied to assess it. Because the process of ME is not a one-time execution workflow but a long-accumulated management framework, the evaluation must be carried out based on historical information. Thereby, a case library is also a fundamental element in evaluation to quantify the quality of the ME process and so as to form a hybrid evaluation mechanism for fast ME evaluation.

Research topics related to evaluation also include quantitative analysis of the complexity and uncertainties risk analysis and control of ME processes, quantitative measurement of model lifecycle quality and cost, etc.

With the combination of model lifecycle evaluation and ME process evaluation, a comprehensive evaluation scheme can be drawn as shown in Fig. 2.6 to guide the further optimization and calibration targeted to the whole ME framework.

2.4.7.3 Model Maturity Definition and Evaluation

The maturity of a model is a very important index for model composition, sharing, and reuse. Maturity definition of a model is not an easy job since different models have different features, different application requirements, and different execution environments. Model maturity will be a comprehensive index related to

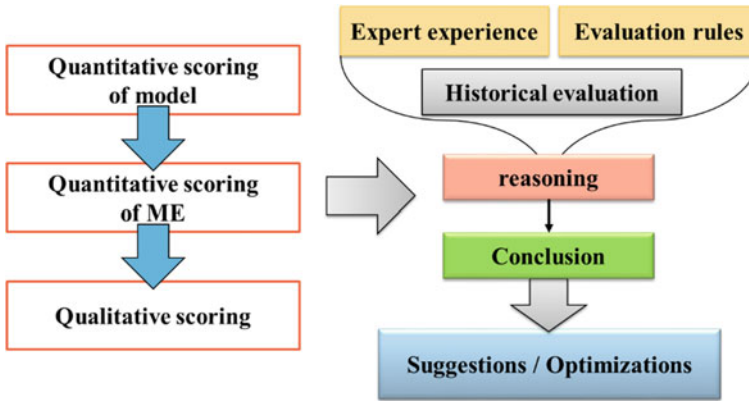


Fig. 2.6 Comprehensive evaluation scheme

multi-dimensional features. Research effort needs to be made on definition and evaluation of model maturity.

2.4.8 Supporting Technologies

The supporting technologies for the implementation of ME primarily consist of the transparent visualization ways for model lifecycle and operational platform to enable fundamental execution of activities involved in the whole process of model engineering, as illustrated in Fig. 2.7.

2.5 Model Curation for Repository Integration: Discovery, Composition, and Reuse

Bernard P. Zeigler, Andreas Tolk, Tuncer Ören

Within the context of model management (Sect. 2.4), we note that a large backlog of legacy simulation models has accumulated over the years that is potentially exploitable for reuse in solving newly arising problems at relatively low cost and quick turnaround. Unfortunately, although organized databases of simulation models were the subject of early research interest [42], much of this source of simulation model knowledge remains untapped. This is due to the fact that legacy models were not developed with later reuse in mind for unforeseen applications. Even today, simulation models are still developed for specific scenarios with built-in organization-specific data and domain-specific (often unstated) assumptions. This renders them hard to apply to evaluation of the effectiveness of newly proposed strategies, designs, and plans. Consequently, interpretation of model

specification of the desired component), and *adaptation* (making changes to a model to meet reuse requirements).

On the foundation the *Theory of Modeling and Simulation (TMS)* (Sect. 1.3) and Modeling and Simulation Framework (MSF) (Sect. 1.4), [47] formalized the duality between simulation models and *experimental frames* (EF) to support frame applicability to models in the service of composability and reuse. They defined a methodology to capture the *observable/objectives, assumptions, and constraints* (OOAC) associated with a given frame and that a model has to satisfy. To enable partial satisfaction, a more relaxed version of full applicability, it was formulated as a conjunction of satisfaction sub-functions (for the OOAC components, respectively) and an interval estimating confidence in the applicability [48].

The formalization of experimental frames (EF) was extended by Traore and Muzy [49] as a basis for specifying the context in which simulation models are built. Such formalization enables development of formal methods for verification of models' consistency, composability, reuse, and validity. Cheon et al. [50] extended the concepts of model matching and the role of context to further support ontology-based model discovery, noting that much work is left to reduce the proposed concepts to working mechanisms.

Zeigler and Nutaro [51] further extended the TMS from the “modeling in the large” perspective of Zeigler [52]. The extension enables suitable EFs to computationally represent given IUs. In this theory, a model developed for an application is expected to be valid in one or more EFs associated with the IU that formalizes that application. In the context of evaluating candidate models, this enables users to select candidates for composition or integration that are best suited for a given IU. In effect, the EF associated with an IU acts as a key to inform search for the models whose EFs best fit it.

This foundation allows us to discuss how to *curate* (annotate based on well-founded theory, [53] simulation models so that they can more easily discovered from a model repository given analytical objectives. From a historical perspective, curation is a form of symbolic processing of simulation models [12] within a larger scope of model-based activity that was envisioned early in the development of simulation languages and concepts [12].

This breaks down into three main considerations:

- (1) Employ the system theory-based entities and relations identified by TMS to develop a knowledge structure that covers the most commonly employed modes of expression (mathematical, statistical, algorithmic) and types of formal specifications and suggests derivable mathematical specifications and useful theorems to support model curation;
- (2) Develop a model-matching process and workflow based on the developed knowledge structure to discover models that match analytical objectives formulated in a manner attuned to the curation structure;
- (3) Augment the TMS-based knowledge structure to capture linkages and attributes necessary to validate the correctness of the curation with respect to the modeler's intent.

We assume that the entities in a model repository will be found in an uncurated form that we will refer to as “Model Packages”. The questions then concern how to design an approach to curation of such packages that can be executed with available computational resources and converts them into a form that satisfies the TMS specifications for decomposition into model, simulator, experimental frame, and associated relations.

1. Decomposing a model package into its MSF elements

Having assumed that model packages constitute our starting entities, we must develop an approach to representing a model package by its elements from the MSF. Basically, this requires that we separate the model from the simulator and experimental frame. As a start, we consider the following characterization:

- A *model* is (collectively) that part of parts of the package that relates to the real system of interest, e.g., it might make a claim about how some mechanism in the system works. A model can be considered as a set of instructions that when executed are intended to replicate or predict the real system’s behavior.
- A *simulator* is the code sections that directly, or indirectly, execute the model’s instructions and generate its behavior.
- An *experimental frame* is constituted by the sections of code that relate to the conditions under which the model is intended to simulated, i.e., the inputs, controls, and expected outputs that specify the realm of system behavior of interest.

2. Identifying a model’s underlying formalism and level of system specification

Having identified the model within a model package, critical to its curation for eventual reuse, and composition is the proper designation of the software from the perspective of the predictability of its behavior in new compositions. The system specification hierarchy of TMS provides an orderly way of establishing relationships between system descriptions as well as presenting and working with such relationships. DEVS compliance is necessary to assign a model package to the right level of system specification and to curate it for accurate repository discovery using the SMRC architecture. Wymore and Bahill [54] relate the incident in which a virtual reality simulator for troop movement in challenging terrain was reused to depict mobs of kangaroos in the same environment with superficial adaptations. This proved successful until a demonstration revealed that such ‘kangaroos’ also could regroup and launch missiles. Here reuse was based on behavioral equivalence (with respect to movement) but neglected to check equivalence at the more revealing state equivalence level. The system specification hierarchy (Sect. 1.4.1) provides an orderly way of establishing relationships between system descriptions as well as presenting and working with such relationships.

Table 2.14 System specification levels and types framework for model package characterization [55]

System specification/level of specification	Differential equation system specification (DESS)	Discrete time system specification (DTSS)	Discrete event system specification (DEVS)
Observation frame	✓	✓	✓
I/O behavior	✓	✓	✓
LO function	✓	✓	✓
State transition	✓	✓	✓
Coupled component	✓	✓	✓

As in Table 2.14, orthogonal to the level at which a system is specified is the subclass of systems in which the model resides where the most common subclasses are spanned by the modeling formalisms shown in the table. Thus, curation entails assigning an incoming model to the correct cell in the table, and there are two corresponding activities:

- *Determine the level of system specification in which a model resides:* This requires characterization of the levels of system specification in an operational form that enables examination of the documentation to uncover whether the model is presented as data at the I/O behavior level, behavior generation instructions in the form of a state diagram, or in the form of interacting components, etc.
- *Characterize the system specification formalism of model:* This requires identification of the models expressed in types of simulation languages as instances of the basic types of system specification: DESS, DTSS, and DEVS. The table shows that such system specifications can occur at any of the levels of specification.

Together, these processes implement the assignment of a model to a pair (system specification level, system specification formalism) which constitutes essential knowledge upon which to base the possibilities for reuse and composition.

To add models to the repository, we must develop an approach to characterizing the model package's intended use. We must sufficiently refine the concept of analytical objective and relate it to the operational simulation consequences expressed in the experimental frame. Specifically, the IU requires.

- Narrative description of the analysis problem and tasks or processes the IU stakeholder user performs
- Specification of the outputs that the stakeholder user will employ in the process
- Special conditions of experimentation or runs to produce outputs usable in the stakeholder's process

- Threshold acceptance or goal goodness criteria for each IU.

Recommendation of best configuration fulfilling input objectives in the form of IUs is formulated in the form of model matching. The model-matching concept based on IUs requires developing concepts, and associated software tools enable locating the most suitable model in the repository for a given analytical objective. Here TMS provides a set of relations between experimental frames and models that provide a basis for such matching.

The relations outlined in Table 1.4 in Sect. 1.4.4.1 must be implemented in algorithmic form so that the given EF can be tested for derivability against the EFs of models in the repository and accommodation by the model with best matching EF confirmed to enable simulation. The duality between simulation models and experimental frames for the algorithmic implementation given by Traore and Zeigler [47] can be employed to capture the observable/objectives, assumptions, and constraints (OOAC) associated with a given frame and that a model has to satisfy. In addition, more advanced features need to be implemented such as partial match satisfaction and confidence interval estimation of signature matching and specification matching [56] to build an efficient model frame/applicability prover.

1. SES Support for Model Repositories

The System Entity Structure (SES) (Sect. 1.5) is used to organize the family of all possible configurations of a system. The SES enables the development of model repositories of reusable M&S components. SES model repository components are composable and executable to fulfill the objectives of stakeholders' IUs. Reusability support methodology differs in a major respect from most simulation environments in its use of the SES to support composability (Sect. 2.4). The composability feature of SES results in significant reduction in time to develop models for new objectives that can only be emulated with the use of ad hoc configuration scripts that are needed when using DEVS alone SMRC facilities for objective-driven SES pruning guide the user and automate the configuration process to achieve an executable SMRC composition best suited to fulfill the stakeholder's objectives for the IU [57]. The SES supports model repositories using the suites of models in the MS4 Me environment [58]. In a suite of models, each component SES represents a family of models that can be pruned and transformed to execute in a simulation. Component SESs can be merged to a new SES with the same compositional properties. This functionality leads to the concept of a repository of models. SES supports families of models for combinatorial generation of architectural alternatives for exploration and optimization. As an ontological framework, the SES supports composition of models drawn from one or more model repositories. Operations on SES objects such as merging ease development by maintaining coherence of shared constructs among modifiable components. Merging enables divide-and-conquer component-based development of suites of models.

Figure 2.8 depicts a set of intersecting SESs representing a suite of related families of models concerning global warming. The families, and the questions they address, include:

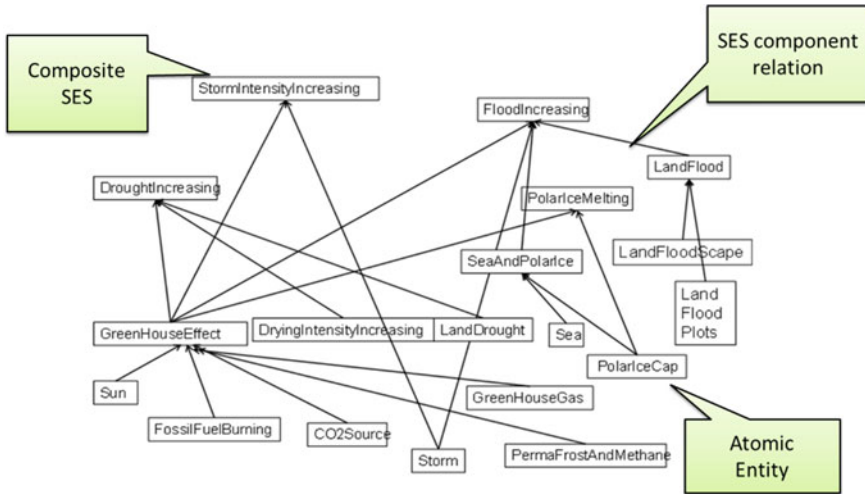


Fig. 2.8 Composition structure of SESs representing global warming families of models

- *Greenhouse effect model family*: What is the greenhouse effect that causes warming of the earth?
- *Polar ice melting model family*: How does global warming cause increased melting of the polar ice cap?
- *Permafrost melting model family*: How does the warming of the permafrost contribute to ever increasing global warming?
- *Sea level rising model family*: How does the global warming contribute to the rising of the sea level?
- *Storm intensity increasing model family*: How does the global warming contribute to the increasing intensity of storms?
- *Flood increasing model family*: How does the global warming contribute to the increasing incidence of floods?
- *Drought increasing model family*: How does the global warming contribute to the increasing incidence of droughts?

In such a suite, there are SESs that are “components of other SESs. This use of the term ‘components’ transfers the “component of” concept from its use in component-based model construction [59] to the domain of SES construction. Thus, an SES is a component of another SES in the sense that the models the first SES generates are components of models generated by the second SES. The operation of composing DEVS models to create a coupled model is mirrored by the merging operation for composing SESs.

As illustrated in Fig. 2.8, these SESs form a set that is related by a composition relation. Here an arrow indicates composition, i.e., an SES is composed of the SESs and atomic entities sending arrows to it. For example, the GreenHouseEffect SES is composed only of atomic entities, while it is a component in many other SESs,

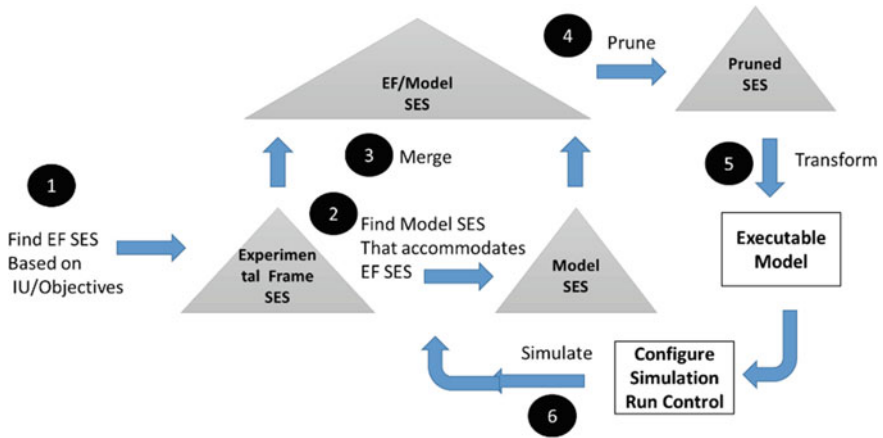


Fig. 2.9 Merging, pruning, and transforming SESs

reflecting the case that greenhouse gases are a cause of the related climate change phenomena.

The process for developing, merging, pruning, and transforming an SES is illustrated in Fig. 2.9. The workflow supporting the development of an SES representing a new family of models in an existing suite of models starts with a top-down search starts with looking for existing models that can be used as components. Identification of candidate models can be supported by the objectives and intended uses that motivate the development of the family as discussed in Section x. In Step 1, an experimental frame in the form of an SES is formulated either fresh or as needed from existing EF components. In Step 2, existing family of models is sought to accommodate the EF SES. Such existing models include families of models generated by SESs as well as atomic models in the repository. An existing family of models generated by an SES becomes a component SES when its name appears as a leaf entity in the SES under development. In a later pruning process, this component SES will be merged into the target SES. For needed components that are currently not available in the suite of models, the same procedure applies, with the additional task of recursively developing the components in the same manner as the target. Of course, one may decide at any point that is better to develop a needed component as an atomic model rather than as one generated by an SES. In Step 3, the EF and model SESs are merged to form a simulation model family for execution. Note that before pruning, the SES components of the resultant SES are merged (recursively) to give the merged version of the resultant SES. In Steps 4 and 5, the resultant SES is pruned and transformed to a hierarchical coupled model. For example, to create the merged SES for FloodIncreasing, the unmerged SES is merged with the component SESs GreenHouseEffect, SeanAndPolarIce, and LandFlood, where the latter is merged from components, LandFloodScape and LandFloodPlots. In Step 6, the executable model is configured by setting parameter values that are not set through the pruning choices and simulation results are

obtained [57] The cycle is repeated if necessary to converge to desired results (methodology for conducting such iterative experiments is discussed in other literature such as [29] (Fig. 2.9).

The concept of repository of models extends the support for developing suites of models by using Web Services backed by cloud technology. For collaborative team developers, it provides the composition and integration facilities based on the SES described above. Going beyond support for sharing, it provides advanced support for browsing the SES structures of models aimed at enabling comprehension of model content and functionality. Such comprehension is necessary to realistically enable a developer to acquire and reuse a simulation model developed by someone else. Further research is needed to develop the most efficient and effective ways of generating views of SES entities, relationships, and variables. After construction, the suite of models can be hosted in a cloud-based repository of models as a basis for collaborative model development.

1. Further Research Needed in a Wider Context

A model-model coupling interface concept can be a prerequisite for many composition-related features [60, 61]. Acquiring additional interface information was found to provide a foundation for desired features such as (1) unit testing, (2) automatic unit conversion, and (3) the automatic detection of specialization and multi-aspect pruning options. Interfaces make the creation and management of unit tests easier for modelers. Unit conversion can be automated by inspecting the units of interfaces. The unit conversion feature includes the conversion of multi-dimensional units such as coordinate transformation (e.g., Polar to Cartesian). The SES concept can be expanded to allow for pruning options to include models that satisfy certain interfaces.

A significant technical challenge will be to characterize the composability of existing models, developed for use in external frameworks, as DEVS components. In addition to the theoretic elements of composability expressed in the SMRC architecture, there are the technical and computing elements of composability that define the compute architecture and information exchange protocols that allow them to work together.

The DEVS-DMF framework developed by Kewley et al. [62] is a DEVS implementation based on microservices that enables location transparency and affords the use of web and cloud-based technologies for integration. In short, DEVS models can be microservices running on any computational infrastructure and any location, integrated via a variety of widely used information exchange protocols such as HTTP, messaging systems, or websockets. Kewley et al. [62] developed soldier models that can be wrapped to work in a variety of simulation frameworks by breaking DEVS models into stateless (transition functions, output functions) and stateful (state variables) components in a distributed environment using the actor model of computation.

MS4 Me and DEVS integrating technologies to capture the requirements and modeling and simulation as a service (MSaaS) are a promising approach to improve efficiency in simulation use and better utilization of valuable resources [63, 64].

Experiments are being conducted to investigate operational benefits of MSaaS repositories to improve awareness of existing resources and to offer workflow support for key MSaaS activities such as resource discovery, exercise preparation and setup, service composition, and finally deployment and execution.

Finally, it is interesting to place model curation within the broader framework of modeling and simulation as an integrative factor in bringing together multiple disciplines to address complex problems. Addressing the today's challenging complex problems, such as the recent response to the pandemic, or the increasing social need for equality in many daily domains, requires the collaboration of experts from multiple disciplines. Simulation studies are increasingly conducted by teams of multidisciplinary, interdisciplinary, and transdisciplinary researchers and practitioners, who apply theories, methods, and tools from their respective disciplines toward a common solution. To take full advantage of curation, a formal alignment of conceptual approaches is needed. Tolk et al. [65] present a conceptual framework for hybrid approaches generally applicable to all kinds of computational support of research presenting a framework that supports the collaboration of research efforts from multiple disciplines. This transdisciplinarity-enabling methodology allows such efforts to grow into transdisciplinary research. Research is needed to further evaluate the degree to which these concepts can be applied to allow the utilization of curations beyond the boundaries of the discipline that created the original model so as to enhance the model's applicability in such a broader transdisciplinary context. These approaches can extend systems engineering approaches, such as described in [66].

2.6 Model-Based Simulation

Greg Zacharewicz

This section presents most commonly used model-based and model-driven approaches (MB-MD) to reach a simulation model. Indeed, all major research initiatives agree that the problem of inconsistency in system and simulation design can be partly solved by early unambiguous descriptions (i.e., semiformal or formal model) of the system [67]. We identify and detail the main problems related to the need of early modeling, discuss current solutions, approaches, and frameworks, and present the relative obstacles that need to be overcome to implement simulation.

As pioneer contribution, authors of [3] described a simulation program that was dichotomized for the first time, and lines of simulation program code were identified as representing the model and different elements of the experimental conditions.

The evolution from document or speech-based software engineering to model-based software engineering [68] concurs with the model-based approach already successfully applied to simulation and systems engineering. The unified modeling language (UML), also developed in the 1990s [69], has opened new

horizons in the field of software-intensive systems engineering. In system domain, according to INCOSE, MBSE is part of a long-term trend toward model-centric approaches adopted by other engineering disciplines, including mechanical, electrical, and software. Then, simulation-based analysis is an effective technique to support verification and validation of complex systems throughout their lifecycle, and it is another challenge in the model-driven cycle.

According to Tuncer Ören, the motivation for model-based approaches in software engineering is to improve productivity through the generation of software artifacts, including source code through the transformation and progressive refinement of models [70].

2.6.1 Model-Based/driven Approaches

There are several types of model-based/driven approaches in system and software engineering. We introduced a non-exclusive categorization of the model-based approaches used by engineering community frequently extended:

- Model-based engineering (**MBE**) utilizes model-driven practices pragmatically, not necessarily in an integrated fashion, in various steps of the engineering process. While models are important in MBE, they do not necessarily drive the development process. In this sense, all model-driven processes are regarded as model based.
- Model-based system engineering (**MBSE**) aims to facilitate system description understanding by moving from a classic documentary, textual approach, used for several years, to a model approach. The companies in charge of the implementation thus receive a model, a conceptual/formal/visual approach that is more efficient than the textual approach. Current activities of the MBSE initiative are sponsored by the International Council on Systems Engineering (INCOSE) [71].
- Model-driven development (**MDD**) proposes a paradigm that utilizes models as the primary artifacts and redefines the implementation as (semi) automatic generation from the models. The process relies on the use of models and the systematic production and transformation of models.
- Model-driven engineering (**MDE**) expands MDD to cover all engineering process areas with a focus on developing metamodels to facilitate automated transformations.
- Model-driven architecture (**MDA**) is the particular vision of MDD proposed by the Object Management Group (OMG). It is originally focused on software development.
- Model integrated computing (**MIC**) refines MDD approaches and provides an open integration framework to support formal analysis tools, verification techniques, and model transformations in the development process [72].
- Model-driven systems engineering (**MDSE**) defined by Mittal and Martin [73] for system of systems utilizes the benefits of both MBSE and MDE

- Model-driven interoperability (**MDI**) methodology that was realized in the frame of the Task Group 2 (TG2) of INTEROP-NoE [74] to tackle interoperability. The goal is to tackle the interoperability problem at each abstraction level defined in MDA and to use model transformation techniques to link horizontally to ensure interoperability of models of collaborating enterprises at each level.
- Model-driven system engineering architecture (**MDSEA**) [75] fills a gap in other methods by considering resource categories. It separates the IT, human organization, and physical means resources. This separation allows to better model and control their interaction with the system to be specified.

These MB-MD approaches are tackling model-oriented needs of users, but the behavioral aspect regarding time is forgotten. Indeed, these approaches are not situating neither involving explicitly and natively simulation in the approaches. A simulation-driven engineering approach can be an important resource for all researchers and practitioners involved in system or software engineering, who may be interested in adopting MDE principles when developing systems. But the development of a simulation is by itself a challenging task in terms of effort, guidelines, and required know-how.

2.6.2 Simulation-Based/Driven Approaches

Model-driven simulation engineering is hot topic in the 2020s. As one state of the art, authors [76] provides a comprehensive review of distributed simulation (DS) from the perspective of model-driven engineering (MDE), illustrating how MDE affects the overall lifecycle of the simulation development process. They describe a road map for building a DS system in accordance with the MDE perspective and a technical framework for the development of conceptual models. They present a focus on federate (simulation environment) architectures, detailing a practical approach to the design of federations (i.e., simulation member design). They discuss the main activities related to scenario management in DS and explore the process of MDE-based implementation, integration, and testing.

Moreover, some more technical contributions have considered that project methodology or lifecycle is crucial because the inherently distributed nature of complex systems makes the use of heterogeneous simulation approaches hard to set up. For that purpose, IEEE HLA standard [77] has proposed FEderation DEvelopment Process (FEDEP) and now Distributed Simulation Engineering and Execution Process (DSEEP) to develop distributed simulations with keeping the objective of model reuse and interoperability. Automated model-driven engineering principles and standards to ease the development of distributed simulations. Authors [78] approach is framed around the development process defined by the DSEEP standard, as applied to distributed simulations based on the high-level architecture (HLA), and is focused on a chain of automated model transformations. A case study is used in the tutorial to illustrate an example application of the

proposed model-driven approach to the development of an HLA-based distributed simulation of a space system.

Another important point is about the fact that many projects start from existing and legacy systems approaches to simulation evolution and modernization, including architecture-driven modernization for simulation modernization and, also, potential synergies and interoperability between other systems while developing and running such as agent, DS, and MDE methodologies, suggesting avenues for future research at the intersection of these three fields.

2.6.3 Models Simulations-Based/Driven Approaches

As stated in the previous approaches, model development is mainly IT directed and the models are already designed with IT goal, not always centralizing the users view and requirements. Moreover, most of the MD approaches presented do not propose and support specific modeling languages. Another drawback is that model transformation remains limited to certain categories of models as presented by Den Haan [79]. Also, the modeler is not guided to delimit the subpart of the system to be computerized. Silingas [80] stated that a tool to identify and delimitate the subpart of the model to be considered at a down level modeling would be valuable to prepare transformation. In addition, the transformation is not fully automatic; the model generated from model transformation is typically an abstract structure with slight content coming from the information collected in the source model. The transformation to lower-level models requires systematic user enrichment.

As one recent tentative of convergence and inclusion of previous MB-MD approaches, authors [81] introduced Fig. 2.1 with the goal to gather best practices in MD-MB. Efficient model matchings ensure both horizontal interoperability (between blocks) and vertical interoperability model transformation to ensure interoperability between business people (dotted zone) and the IT department (dashed zone). As well in this figure, authors introduced the simulation as a systematic step in MD-MB approaches. In consequence, one raw is dedicated to simulation in the figure (yellow blocks). Authors believed that simulation is a mandatory step in the model-driven approach. It is confirmed by authors [70] that stated that any complex system study (design, analysis, or control) cannot be conducted without considering simulation-based techniques. They state that simulation-based approaches in any discipline are a rational way to enhance engineering performance in an effective way. As recap, simulation is proposed to be used as a middle layer in system development for behavior prediction, performance analysis, analysis of alternatives, sensitivity analysis, engineering design test, virtual prototyping, planning, acquisition, and proof of concept. To complete this approach, the framework includes a model discovery from legacy systems flow, and it is as well open to human and organization thanks social media links and human interfaces (Fig. 2.10).

Furthermore, although the MD-MB field is becoming increasingly aware of simulation, studies by Ören and his colleagues conclude that the role of simulation

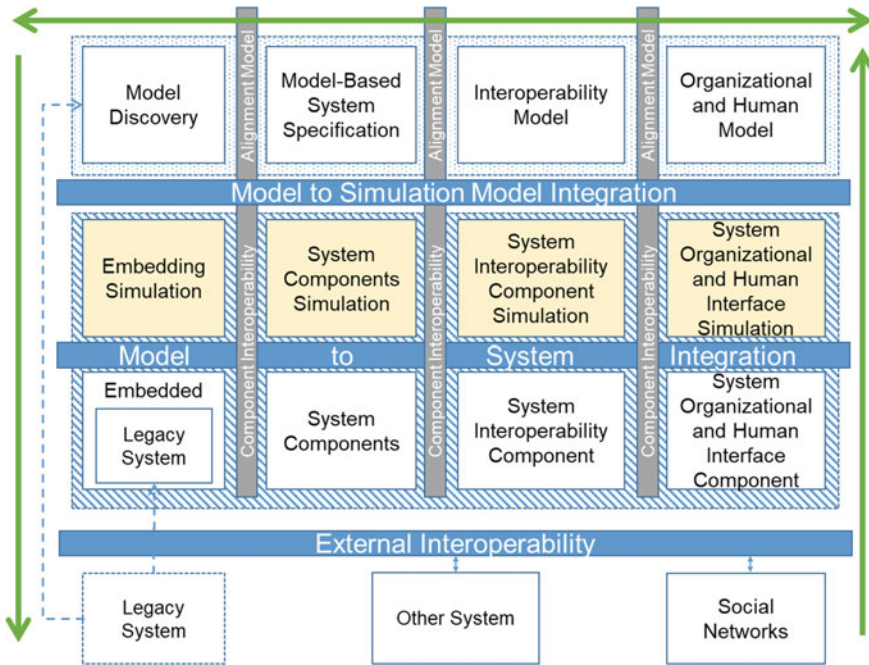


Fig. 2.10 Models simulation-based/driven conceptual framework

is still not systematically and sufficiently taken into account methodologically. It needs to be promoted through education in many disciplines so that future professionals in these disciplines can be better equipped for their profession. They propose to involve simulation—even non-computer based—in the teaching of young children, so that future generations will be better prepared to have better thinking skills.

2.7 Transient Elimination in Simulation

Mohammad S. Obaidat, Balqies Sadoun

The practice of analyzing simulation outcomes is a crucial one since without such an analysis we will not have good confidence that the simulator is accurate and close to reality to a reasonable degree. Every time a model requires to be realized for a real-time application, we have to test if the functionality of the model meets our needs and meets the anticipated objectives. This article gives a review on the techniques used to eliminate transient results in simulation outcomes as transient results may affect the credibility of simulation results if they are not removed. It also sheds some light of criteria that are used to stop/end simulation in order to save resources without affecting credibility of outcomes.

2.7.1 Introduction

In almost all simulation techniques, we are interested in the performance of the simulation model at steady-state situations, which means that we should remove the initial part of results from the final outcomes so that we can have accurate conclusions to make decisions. This initial portion of simulation results is frequently termed the transient part/state. Recognizing the completion of transient state is often called transient state elimination or removal. The key problem about transient state is that it is not easy to delineate the interval of the transient state and where it essentially ends [82–103].

In this article, we shed some light on the major approaches used to eliminate transient results in simulation outcomes as well as schemes to end simulation properly so that we get safe resources, but at same time get credible simulation results.

2.7.2 Techniques to Eliminate Transient Results

We, here, review the major techniques used for transient removals as well as their features and domains. Almost all of these schemes are heuristic techniques that are usually applied for transient elimination [82–86].

1. **Long runs methodology.** In this technique, the simulator is executed for an extended length to the level that the presence of initial states will be insignificant or will not affect the outcome. This technique appears to be easy; however, it misuses the computing time and related resources. Besides, it is difficult to identify the duration of the simulation run that can lessen the effects of initial outcomes [82–89] (Fig. 2.11).
2. **Batch means methodology.** In this scheme, the simulation is executed for a long time, and then it is divided into several equal durations. Each part or division is named a batch. Each mean observation in each batch is called a batch mean. This is the reason that leads to call this technique the batch means [82–87].
 - (a) A batch average is calculated for every batch.
 - (b) The overall average is then figured out.
 - (c) Then, the variance of the batch means is then determined.

These (a) and (c) steps are repeated by changing the size of each batch ‘n’. A chart is then drawn with variance for a range of batch sizes n. While the variance gets decreasing, the related value of ‘n’ is described as the length of the transient period/time [82–85].

3. **Truncation procedure.** In this scheme, we assume that the changeability or variability in the transient state is greater than that in the steady state, which is typically a rational hypothesis. This method determines the extent of the

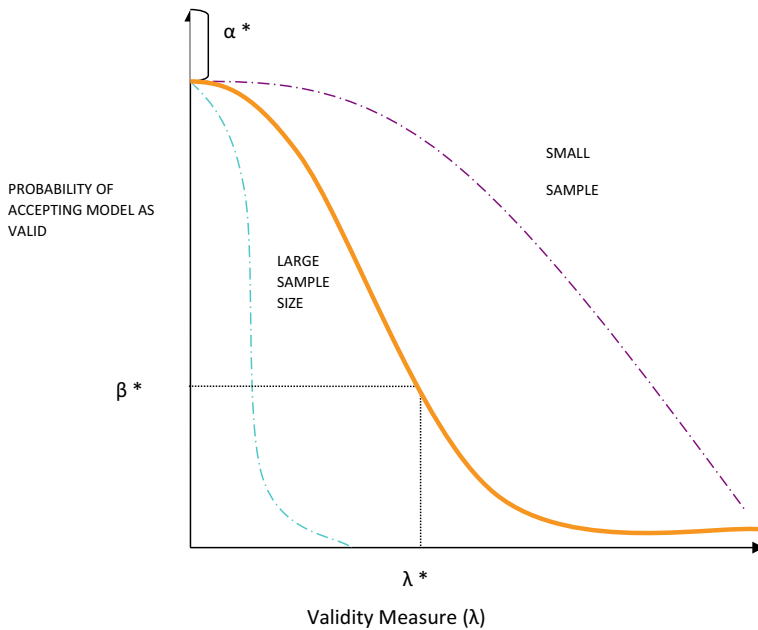


Fig. 2.11 Operating characteristic curves

changeability like the greatest and smallest number of observations. If we work out these observations on a graph, then we will be able to see that the curve turns out to be stable as the simulation moves to the steady state [82–89].

4. **Good initialization.** In this method, the simulation program is commenced in a state that is neighboring to the projected steady state, which is generally non-zero. The length of the transient phase is decreased, thereby having a slight influence on the total performance results. For instance, the normal queue size in the input or output buffer of an asynchronous transfer switch model is non-zero; therefore, in simulation we initialize the queue with a representative value that is found from experience and past record.
5. **Initial data deletion.** In this scheme, some of the early observations from the sample are eliminated after complete analysis. During the entire steady state, the mean does not vary, even though the observations are removed. Nevertheless, the mean can change even during a steady state due to randomness of the observations. The outcome here can be minimized by averaging throughout a number of replications [82–91].

If we suppose that we have ‘ m ’ replication with size ‘ n ’ for each and x_{ij} is the j th observation in the i th iteration where j changes from 1 to n along the time axis, whereas i varies from 1 to m along replications axis, this methodology can be summarized by the following steps:

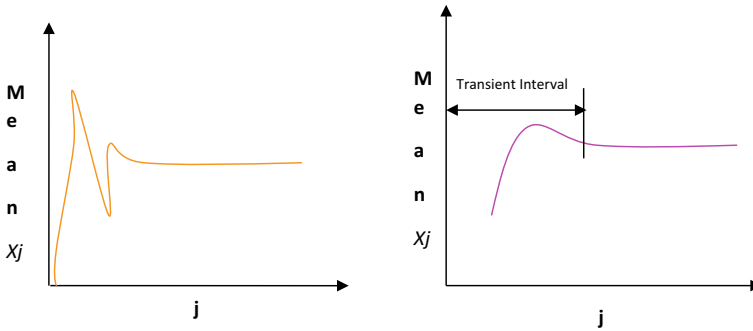


Fig. 2.12 Moving average of independent replications

- (a) By averaging across the repetitions/repetitions, a mean trajectory is obtained.
- (b) Then the total (general) mean is attained.
- (c) If we suppose that the interval/length of the transient state is l , then *the* whole (overall) mean is obtained by removing the first l observations from the mean path or curve/trajectory.
- (d) This, the relative change in the overall mean is calculated.
- (e) Next, we do again the steps by changing the values of l from 1 to $(n - 1)$. The charts of the common average and relative change are plotted in order to see that after a particular value of l , the relative variation plot settles down. This point is termed the knee, which essentially provides the length of the transient period and exhibits the conclusion of the transient state [82–91].

6. Moving mean of separate replications. This scheme has good similarity with the initial data deletion procedure excluding that the mean in this method is found out over a moving time period window rather than by computing the overall mean.

If we suppose that we have ‘ m ’ replications with size ‘ n ’ for each. Then, let us denote by x_{ij} the j th observation in the i th iteration/repetition where j varies from 1 to n across the time axis while i changes from 1 to m through the replications axis. The phases shown below summarize this technique:

1. The average trajectory is attained by averaging the repetitions/replications.
2. Then, we plot a path for the moving average of the consecutive $2k + 1$ values where k signifies the moving time interval window.
 - (b) Next, we repeat phase 2 for various values of $k = 2, 3, \dots$, until a smooth plot is achieved.
 - (c) Finally, the interval of the transient interval is attained by locating the knee on this curve.

Figure 2.12 depicts two distinctive trajectories of moving averages. The curve of the second trajectory is smooth, and thus, identifying the knee would be easy.

2.7.3 Stopping Criteria for Simulations

In simulation modeling, it is essential to simulate the system under study for sufficient length of time. If the simulation time is short, then accuracy and credibility of the results are in doubt. However, if the simulation time is lengthy, then we mainly waste the computation power and used resources.

We can identify three chief methods that permit the simulation to be executed until the required confidence interval is obtained. These are independent replications, rebirth, and batch means techniques [82–97].

1. **Independent replications scheme.** In this scheme, simulation is reiterated with a different seed values so as to obtain various replications. Now, if we have m replications that are performed of size $n + n'$, where n' represents the transient interval length, then we remove the first n_0 observations and implement the following phases [83–90]:
 2. For every repetition or replication, the mean is calculated.
 3. The total mean is then computed for all of the replications.
 4. The variance of these replicate averages is then computed.
 5. The confidence interval is achieved by the summation of the overall mean and the variance as shown below: overall mean $\pm Z_{1-\alpha} \text{Var}(x') = \bar{x}'' \pm Z_{1-\alpha} \text{Var}(x')$, where $Z_{1-\alpha}$ is obtained from special tables; quantile unit normal variate table.

The width of the confidence interval is inversely proportional to square root of mn . In this, we can get a narrower confidence interval by either enlarging ‘ m ’ or ‘ n ’ [82–87].

2. **Rebirth technique.** A rebirth or regeneration point is defined as the instant at which the system enters into an independent stage. The time between two such points is named as rebirth or generation cycle.

Now, let us suppose that we have a regenerative simulation that encompasses m phases with sizes $N_1, N_2, N_3 \dots N_m$. Thus, the confidence interval can be found by following the steps below:

1. The cycle totals are calculated, and the overall mean is established.
2. Next, the differences between expected and noticeable/observable cycle sums are figured out.
3. Finally, the variance for these differences is also computed along with the mean cycle interval.

The confidence interval is found out by exploiting the overall average, variance, and the average cycle length. It is noted that the rebirth technique does not impose the transient interval to be excluded. This scheme has some shortcomings: (a) Majority of the variance reduction schemes cannot be employed as the length of the cycles is not fixed and cannot be projected, (b) the length of cycles is irregular,

(c) the expected values for means and variances are not equal to the size that is being assessed, and (d) it is hard to locate the rebirth points [82–86].

3. **Batch averages/means.** In this method, the full interval of the simulation length is partitioned into m batches of similar size n by discarding the transient interval period. The long run of $(n + n')$ is apportioned into m batches by eliminating the transient interval, where n' denotes the transient interval length and the following phases can be performed:
 1. For each batch, the average/mean is determined.
 2. Next and after finding the average for all batches, the overall average is then computed.
 3. Finally, the variance of the batch means is then established.

Thus, the confidence interval is then obtained as the total of the entire mean and the variance. Size of the confidence interval is conversely proportional to square root of mn . Given this, we can say a thinner confidence interval can be realized by either increasing ‘ n ’ or ‘ m ’ values [82–91].

2.7.4 Conclusion

We reviewed the key techniques to remove transient results of simulation outcomes in order to reduce their effects on credibility of simulation model and its validity and accuracy. The process is challenging, and most proposed approaches are heuristic. We also studied the chief criteria to stop simulation and decide when it is time to do so in order to save computation power and resources.

References

1. Ören TI (2009) Modeling and simulation: a comprehensive and integrative view. In: Yilmaz L, Ören TI (eds) *Agent-directed simulation and systems engineering*. Wiley Series in Systems Engineering and Management, Wiley-Berlin, Germany, pp 3–36. <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.462.5002&rep=rep1&type=pdf>. Accessed 2019–10–27
2. Ören TI (2011) Modeling and simulation: big picture. China Lecture 1a (Beijing: Beihang University, School of Automation and Electrical Eng. and Changsha: National Univ. of Defense Technology, System Simulation Lab.) <http://www.site.uottawa.ca/~oren/y/2011/09-bigPicture.pdf>. Accessed 2019–10–27
3. Ören TI, Zeigler BP (1979) Concepts for advanced simulation methodologies. *Simulation* 32 (3):69–82. <https://doi.org/10.1177/003754977903200301>
4. Keller D (2007) Jean Baudrillard. Stanford encyclopedia of philosophy. <https://plato.stanford.edu/entries/ baudrillard/>. Accessed 2019–10–28
5. Dewey J (2010) *Art as experience*. Penguin Putnam Inc., New York
6. UK-SEMS—UK Ministry of defence synthetic environment & modelling & simulation (SEMS) glossary. <https://webarchive.nationalarchives.gov.uk/+/http://www.mod.uk/issues/ simulation/glossary.htm>. Accessed 2019–10–27

7. USA-DoD—DoD M&S Glossary—DoD 5000.59-M. <http://www.acqnotes.com/Attachments/DoD%20M&S%20Glossary%201%20Oct%2011.pdf>. Accessed 2019–10–27
8. Ören TI, Zeigler BP (2012) System theoretic foundations of modeling and simulation: a historic perspective and the legacy of A. Wayne Wymore. Special Issue of Simul Trans SCS (abstract) 88(9), pp 1033–1046. <https://doi.org/10.1177/0037549712450360>
9. Zeigler BP, Muzy A, Kofman E (3rd edn. 2019) *Theory of modeling and simulation: discrete event & iterative system computational foundations*, Elsevier. Please note: Chapters from this latest (3rd) edition can be downloaded from: <https://www.sciencedirect.com/book/9780128133705/theory-of-modeling-and-simulation> or from Researchgate.net (requesting from Alex Muzy or B.P. Zeigler)
10. Wymore AW (1967) *A mathematical theory of systems engineering: the elements*. Wiley, New York, p 353
11. Ören TI (1971) *GEST: general system theory implementor, a combined digital simulation language*. Ph.D. Dissertation, 265 p. University of Arizona, Tucson, AZ
12. Ören TI (1984) *GEST—a modelling and simulation language based on system theoretic concepts*. In: Ören TI, Zeigler BP, Elzas MS (eds) *Simulation and model-based methodologies: an integrative view*. Springer, Heidelberg, Germany, pp 281–335
13. Ören TI (1971) *GEST: a combined digital simulation language for large-scale systems*. In: *Proceedings of the Tokyo 1971 AICA (Association Internationale pour le Calcul Analogique) symposium on simulation of complex systems*, Tokyo, Japan, September 3–7, pp B-1/1–B-1/4
14. Ören TI, Yilmaz L (2015, Invited article) *Awareness-based couplings of intelligent agents and other advanced coupling concepts for M&S*. In: *Proceedings of the 5th international conference on simulation and modeling methodologies, technologies and applications (SIMULTECH'15)*, Colmar, France, 21–23 July 2015, pp 3–12
15. Bacon F (1620) *Novum organum*. (Paperback—2015, CreateSpace Independent Publishing Platform)
16. Ören T (2010) *Modeling and simulation body of knowledge (M&S BoK)*. Obtido em 7
17. Wilson AL, Weatherly RM (1994) *The aggregate level simulation protocol: an evolving system*. In: *Proceedings of winter simulation conference*. IEEE, pp 781–787
18. Irmiler M, Hartl D, Schmidt T, Schuchhardt J, Lach C, Meyer HE, Arabe de Angelis M, Klose J, Beckers J (2008). *An approach to handling and interpretation of ambiguous data in transcriptome and proteome comparisons*. *Proteomics* 8(6):1165–1169
19. Zhou ZH (2007) *Mining ambiguous data with multi-instance multi-label representation*. In: *ADMA*, p 1
20. Heitjan DF, Rubin DB (1991) *Ignorability and coarse data*. *Ann Stat*, 2244–2253
21. Sauerborn GC (1995) *Distributed interactive simulation (DIS) protocol data units (PDUs) implemented into a combat model (A case study of the direct fire module II)*. Army Research Lab Aberdeen Proving Ground Md
22. Jordan MA, Powell N, Phillips CV, Chin CK (1997) *Experimental data analysis: a guide to the selection of simple statistical tests*. *Miner Eng* 10(3):275–286
23. You Y, Sung MY (2008) *Haptic data transmission based on the prediction and compression*. In: *2008 IEEE international conference on communications*. IEEE, pp 1824–1828
24. Morgan M (2012) *The world in the model*. Cambridge University Press
25. Bourton (2020) *Model village*, Bourton-on-the-Water. <https://www.bourtoninfo.com/attractions/the-model-village/>. Accessed June 2020
26. Dunn T (2017) *Model villages (Britain's Heritage Series)*. Amberley Publishing
27. *Model Village*, Wikipedia. https://en.wikipedia.org/wiki/Model_village. Accessed June 2020
28. Wardropper I, Sigel T, Dickerson CD (2012) *Bernini: sculpting in clay*. Metropolitan Museum of Art

29. Kimball (2020) Bernini's terra cotta models exhibited in the Kimball Art Museum, Fort Worth, Texas 2013, <https://www.dallasnews.com/arts-entertainment/architecture/2013/02/14/art-review-a-revelatory-look-into-bernis-studio/>, Accessed June 2020
30. Bucher M, Lallement C, Krummenacher F, Enz C (2004) A MOS transistor model for mixed analog-digital circuit design and simulation. In: Reis R, Jess JAG (eds) Design of system on a chip. Springer, Boston
31. Meerschaert M (2013) Mathematical modeling, 4th edn. Academic Press
32. Enderton H (2001) A mathematical introduction to logic. Academic Press
33. Forrester J (2013) Industrial dynamics. Martino Fine Books
34. Fishwick P (1995) Simulation model design and execution. Building digital worlds. Prentice Hall
35. Zeigler BP, Praehofer H, Kim TG (2000) Theory of modeling and simulation: integrating discrete event and continuous complex dynamic systems. Academic Press
36. Rumbaugh J, Jacobson I, Booch G (2004) Unified modeling language reference manual. The Pearson Higher Education
37. Zeigler BP (1976) Theory of modeling and simulation. Wiley, New York
38. Traoré MK (2019) Unified approaches to modeling. In: Zhang L, Zeigler BP, LaiLi Y (eds) Model engineering for simulation, 1st edn. Academic Press, pp 43–56. <https://doi.org/10.1016/B978-0-12-813543-3.00003-2>. ISBN: 9780128135433
39. Zhang L, Zeigler BP, LaiLi YJ (2019) Model engineering for simulation, (co-volume with TMS3rd Ed.) Elsevier
40. Zhang L (2011) Model engineering for complex system simulation. In: Li J (ed) The 58th CAST forum on new viewpoints and new doctrines. Yunnan, China, October 14–16
41. Zhang L, Shen YW, Zhang XS, Song X, Tao F, Liu Y (2014). The model engineering for complex system simulation. In: The 26th European modeling & simulation symposium (Simulation in Industry), Bordeaux, September 10–12
42. Hitz M, Werthner H, Ören TI (1993) Employing databases for large scale reuse of simulation models. In: Proceedings of 1993 winter simulation conference, pp 544–551
43. Page EH, Opper JM (1999) Observations on the complexity of composable simulation WSC'99. In: 1999 winter simulation conference. In: Proceedings of 'simulation-a bridge
44. Overstreet CM, Nance RE, Balci O (2002) Issues in enhancing model reuse. In: International conference on grand challenges for modeling and simulation, Jan 27–31, San Antonio
45. Balci O et al (2017) Model reuse, composition, and adaptation. In: Fujimoto R, Bock C, Chen W, Page E, Panchal J (eds) Research challenges in modeling and simulation for engineering complex systems. Simulation foundations, methods and applications. Springer, Cham
46. Piersall CH, Grange FE (2014) The necessity of intended use specification for successful modeling and simulation of a system-of-systems CrossTalk
47. Traoré MK (2004) Conditions for model component reuse. Dagstuhl Seminar, 2004. component-based modeling and simulation. In: Barros FJ, Lehmann A, Liggesmeyer P, Verbraeck A, Zeigler BP (eds) Dagstuhl report, January 18–23. Available online via <http://www.dagstuhl.de/04041/Talks/>. Accessed 22 Apr 2004
48. Petty, Mikel D, Weisel EW (2019) Model composition and reuse. In: Zhang L et al (eds) Model engineering for simulation. Elsevier
49. Traoré MK, Muzy A (2006) Capturing the dual relationship between simulation models and their context. Simul Model Pract Theory 14(2):126–142
50. Cheon S, Zeig BP, Kim D (2008) Web service oriented architecture for DEVS model retrieval by system entity structure and segment decomposition. In: 2008 IEEE international conference on information reuse and integration
51. Zeigler BP, Nutaro J (2016) Towards a framework for more robust validation and verification of simulation models for systems of systems. J Defense Model Simul Appl Methodol Technol 13(1):3–16. <https://doi.org/10.1177/1548512914568657>

52. Zeigler B (1984) *Multifaceted modeling and discrete event simulation*. Academic Press, Boston
53. Sesartić A, Fischlin A, Töwe M (2016) Towards narrowing the curation gap—theoretical considerations and lessons learned from decades of practice. *ISPRS Int J Geo-Inf* 5:91. <https://doi.org/10.3390/ijgi5060091>
54. Wymore AW, Bahill AT (2000) When can we safely reuse systems, upgrade systems, or use COTS components? *Syst Eng* 3(2):82–95
55. Zeigler BP, Muzy A, Kofman E (2018) *Theory of modeling and simulation*, 3rd edn. Academic Press, Elsevier
56. Zaremski AM, Wing JM (1995) Signature matching, a tool for using software libraries. *ACM Trans Softw Eng Methodol (TOSEM)* 4(2)
57. Pawletta FH, Deatcu T, Santucci C, Capocchi L (2019) An integrated modeling, simulation and experimentation environment. In: *Python Based On SES/MB And DEVS, SummerSim-SCSC, 2019 July 22–24*, Berlin, Germany
58. Zeigler BP, Chungman S, Doohwan K (2013) System entity structures for suites of simulation. *Models Int J Model Simul Sci Comput*
59. Verbraeck A (2004) Component-based distributed simulations. The way forward? In *Proceedings of the 18th workshop on parallel and distributed simulation (PADS'04)*, pages 141–148
60. Ören TI (2014) Coupling concepts for simulation: a systematic and comprehensive view and advantages with declarative models. *Int J Model Simul Sci Comput (IJMSSC)* 5 (2):1430001–14300017 (article ID: 1430001). <https://doi.org/10.1142/S1793962314300015>
61. Zeigler BP, Keller KN, Ceny J (2020) Supporting the reuse of algorithmic simulation models. *SummerSim*
62. Kewley R, Kester N, McDonnonell J (2016) DEVS distributed modeling framework: a parallel DEVS implementation via microservices. In: *Proceedings of SpringSim*
63. Bocciairelli A, Giglio P (2018) Model transformation services for MSaaS platforms. *SpringSim*
64. MS4 Me, <http://www.ms4systems.com/pages/ms4me.php>
65. Tolk A, Harper A, Mustafee N (2021) Hybrid models as transdisciplinary research enablers. *Eur J Oper Res* 291(3):1075–1090. <https://doi.org/10.1016/j.ejor.2020.10.010>
66. Choi SH, Kang BG, Kim TG (2019) A heuristic procedure for selecting the best feasible design in the presence of stochastic constraints. *IEEE TRANS. SYS., MAN, AND CYBER SYSTEMS* (<https://www.mdpi.com/2079-8954/6/4/40/html>)
67. Chen D, Dumeingts G (2003) Basic concepts and approaches to develop interoperability of enterprise applications. In: *Working conference on virtual enterprises*. Springer, pp 323–330
68. Miller G, Andy E, Ivar J, Henrik J, Allan K, Stephen M, Dave T (2003) Model driven architecture: how far have we come, how far can we go? In: *Companion of the 18th annual ACM SIGPLAN conference on object-oriented programming, systems, languages, and applications. OOPSLA '03. Association for Computing Machinery, New York*, pp 273–274. <https://doi.org/10.1145/949344.949409>
69. Booch G, James R, Ivar J (2005) *Unified modeling language user guide*, (2nd edn). Addison-Wesley Object Technology Series. Addison-Wesley Professional
70. Ören T, Mittal S, Durak U (2018) A shift from model-based to simulation-based paradigm: timeliness and usefulness for many disciplines. *Int J Comput Softw Eng* 3:126. <https://doi.org/10.15344/2456-4451/2018/126>
71. Friedenthal S, Griego R, Sampson M (2007) INCOSE model based systems engineering (MBSE) initiative. In *INCOSE 2007 symposium*, vol 11
72. Brambilla M, Cabot J, Wimmer M (2017) Model-driven software engineering in practice. *Synthesis Lect Softw Eng* 3(1):1–207
73. Kokkonis G, Psannis K, Roumeliotis M (2016) Real time haptic data transferring. In: *2016 wireless days (WD)*. IEEE, pp 1–3







74. Bourey J-P, Reyes Grangel UJI, Doumeingts G, Berre AJ, Pantelopoulos S, Kalamoukas K (2006) Deliverable DTG2. 3 report on model driven. Update 2:3
75. Bazoun H, Gregory Z, Yves D, Hadrien B (2014) SLMToolBox: an implementation of MDSEA for servitisation and enterprise interoperability. In: Enterprise interoperability VI. Springer, pp 101–111
76. Topçu O, Yılmaz L, Oguztüzün H, Durak U (2016) Distributed simulation. In: A model driven engineering approach. Springer
77. Symington S, Morse KL, Mikel P (2000) IEEE standard for modeling and simulation (M&S) high level architecture (HLA)-framework and rules (IEEE Std 1516–2000). IEEE standard for modeling and simulation (M&S) high level architecture (HLA)-federate interface specification, vol 1516. Std
78. Bocciaelli P, D'Ambrogio A, Giglio A, Paglia E (2019) Model-driven distributed simulation engineering. In: 2019 Winter Simulation Conference (WSC). IEEE, pp 75–89
79. Haan JD (2008) 8 reasons why modeldriven approaches (will) fail
80. Silingas D (2013) Why MDA fails: analysis of unsuccessful cases, 8 novembre 2013. <https://www.infoq.com/presentations/mda-case-study/>
81. Zacharewicz G, Diallo S, Ducq Y, Agostinho C, Jardim-Goncalves R, Bazoun H, Wang Z, Doumeingts G (2017) Model-based approaches for interoperability of next generation enterprise information systems: state of the art and future challenges. *IseB* 15(2):229–256. <https://doi.org/10.1007/s10257-016-0317-8>
82. Obaidat MS, Boudriga N (2010) Fundamentals of performance evaluation of computer and telecommunications systems. Wiley
83. Obaidat MS, Nicopolitidis P, Zarai F (eds) (2015) Modeling and simulation of computer networks and systems: methodologies and applications. Elsevier
84. Obaidat MS, Papadimitriou GI (eds) (2003) Applied system simulation: methodologies and applications. Kluwer Academic Publisher
85. Pace DK (2003) Verification, validation and accreditation of simulation models. In: Obaidat MS, Papadimitriou GI (eds) Applied system simulation: methodologies and applications. Springer, pp 487–506
86. Jain R (1991) The art of computer systems performance evaluation. Wiley, NY
87. Sargent RG (2005) Verification and validation of simulation models. In: Proceedings of the 37th winter simulation conference, pp 130–143, December 2005
88. Brady TF, Yellig E (2005) Simulation data mining: a new form of computer simulation output. In: Proceedings of the 2005 winter simulation conference, pp 285–289, December 2005
89. Law AM, Kelton WD (2007) Simulation modeling and analysis, 4th edn. Mc- Graw Hill, New York
90. Ross SM (2006) Simulation. Academic Press, 4th edn.
91. Zeigler BP, Praehofer H, Kim TG (2000) Theory of modeling and simulation. Academic Press
92. Robinson S (2005) A steady state output analysis. In: Proceedings of the 37th winter simulation conference, pp 763–770, December 2005
93. Paul JM, Suppe AJ, Thomas DE (2001) Modeling and simulation of steady state and transient behaviors for emergent SOCS. In: Proceedings of the 14th international symposium on system synthesis, pp 262–267, September 2001
94. Glynn PW (2005) Initial transient problem for steady state output analysis. In: Proceedings of the 2005 winter simulation conference, pp 739–740, December 2005
95. Kelton D (2003) Statistical analysis of simulation output. In: Proceedings of the 2003 winter simulation conference, pp 23–30, December 2003
96. Pooch U, Wall J (1993) Discrete event simulation—a practical approach. CRC Press
97. Rubinstein RY, Melamed B (1998) Modern simulation and modeling. Wiley
98. Banks J, Carson II JS, Nelson BL (2004) Discrete event system simulation. Prentice Hall

99. Obaidat MS (1999) Performance evaluation of computer and telecommunications systems. *Simul J SCS* 72(5):295–303
100. Obaidat MS (2000) Performance evaluation of high performance computing/computers. *J Comput Elec Eng* 26(3–4):181–185
101. Obaidat MS (2001) Performance evaluation of telecommunication systems: models, issues and applications. *Comput Commun J* 34(9):753–756
102. Ould-Khaoua M, Sarbazi-Azad H, Obaidat MS (2005) Performance modeling and evaluation of high-performance parallel and distributed systems. *Perform Eval J North-Holland* 60(1–4):1–4
103. Obaidat MS Advances in performance evaluation of computer and telecommunication systems. *Simul Trans Soc Model Simul J SCS* 83(2):135–137
104. Ören TI (2010) Simulation and reality: the big picture. (Invited paper for the inaugural issue) *International Journal of Modeling, Simulation, and Scientific Computing* (of the Chinese Association for System Simulation - CASS) by the World Scientific Publishing Co. China, vol 1(1), 1–25. <https://doi.org/10.1142/S1793962310000079>
105. Ören TI (2011a) The many facets of simulation through a collection of about 100 definitions. *SCS M&S Mag* 2:2 (April), pp 82–92
106. Ören TI (2011b) A critical review of definitions and about 400 types of modeling and simulation. *SCS M&S Mag* 2:3 (July):142–151
107. Zeigler BP, Muzy A, Kofman E (2018) *Theory of modeling and simulation*, 3rd edn. Academic Press, Elsevier



Simulation as Experimentation

3

Tuncer Ören , Paul K. Davis , Rhys Goldstein , Azam Khan, Laurent Capocchi , Maâmar El-Amine Hamri, Navonil Mustafee , Alison L. Harper , Baocun Hou, Bo Hu Li, and Yang Liu

Abstract

The different types of experimentation and reasons why to use simulation experiments in the various application domains are the topic of this chapter of the SCS M&S Body of Knowledge. It addresses the types of simulation techniques, introduces the simulation of discrete systems using DEVS in detail, and also comprises a section on continuous systems. It concludes with current views on hybrid M&S, real time simulation, and how to cope with comprehensive systems.

Keywords

Modeling and simulation · Types of experimentation · Deep uncertainty · Simulation techniques · Discrete Event Systems Specification (DEVS) · Hybrid simulation · Real-time simulation · Simulation of comprehensive systems

T. Ören (✉)
University of Ottawa, Ottawa, ON, Canada
e-mail: toren@uottawa.ca

P. K. Davis
The RAND Corporation and the Pardee RAND Graduate School, Santa Monica, CA, USA
e-mail: pdavis@rand.org

R. Goldstein
Autodesk Research, Toronto, ON, Canada
e-mail: rhys.goldstein@autodesk.com

A. Khan
Trax.Co, Toronto, ON, Canada
e-mail: azam.khan@trax.co

L. Capocchi
University of Corsica, Corte, France
e-mail: capocchi@univ-corse.fr

3.1 Types of Experimentations

Tuncer Ören

Science as we know it today is no more than 500 years old. It is firmly based on certain rules of procedure that scientists must follow to obtain accurate knowledge. These rules were formulated during a revolution in scientific thinking—the birth of experimental science—and are upheld today by the world’s scientific societies. [1]

The true worth of an experimenter consists in his pursuing not only what he seeks in his experiment, but also what he did not seek. (Claude Bernard)

An experiment is “an operation or procedure carried out under controlled conditions in order to:

- (1) discover an unknown effect or law,
- (2) to test or establish a hypothesis, or
- (3) to illustrate a known law.” (Merriam-Webster-experiment) (reformatted).

Experiments—and specifically replicable experiments—are the backbone of scientific method since Francis Bacon who in 1620 published “*Novum Organum*” to lay down the foundations of the scientific method [2].

There are several types of experiments.

As shown in Table 3.1 two types of experiments can be identified based on the location of the experiments.

Types of experiments performed with living organisms are shown in Table 3.2.

M. E.-A. Hamri
Aix-Marseille University (LSIS), Marseille, France
e-mail: amine.hamri@lsis.org

N. Mustafee
University of Exeter Business School, Exeter, UK
e-mail: N.Mustafee@exeter.ac.uk

A. L. Harper
University of Exeter Medical School, Exeter, UK
e-mail: A.L.Harper@exeter.ac.uk

B. Hou
Midea Cloud Tech Co., Ltd, Jilin, China
e-mail: houbc2@meicloud.com

B. H. Li
Beihang University, Beijing, China
e-mail: bohuli@moon.bjnet.edu.cn

Y. Liu
CASICloud-Tech Co., Ltd., Haidian, China

Table 3.1 Types of experiments based on the location of the experiment

	Criteria	Type of experiment	Also known as
Experiment is performed	In original place of objects	<i>In situ experiment</i>	<i>Field experiment</i>
	In a computer	<i>In silico experiment</i>	<i>Simulated experiment (Computerized experiment)</i>

Table 3.2 Types of experiments performed with living organisms

	Criteria	Type of experiment		Also known as
Experiments performed with living organism	In their normal biological context	<i>In vivo experiment</i>		In the body <i>experiment</i>
	Outside their normal biological context	<i>ex vivo (on a biological sample) experiment</i>	Sample taken from the donor	<i>Lab experiment</i>
			Sample cultivated in a test tube	<i>In vitro experiment</i>

Thought Experiment

“Thought experiments are performed in the imagination. We set up some situation, we observe what happens, and then we try to draw appropriate conclusions. In this way, thought experiments resemble real experiments, except that they are experiments in the mind. The terms ‘thought experiment,’ ‘imaginary experiment,’ and ‘Gedankenexperiment’ are used interchangeably” [3, 4] (Table 3.2).

Seven types of thought experiments are identified more information at (Wikipedia-thought experiment).

- **“Prefactual thought experiments**—speculate on possible future outcomes, given the present, and ask ‘What will be the outcome if event E occurs?’” (Wikipedia-thought experiment).
- **“Counterfactual thought experiments**—speculate on the possible outcomes of a different past; and ask ‘What might have happened if A had happened instead of B?’” (Wikipedia-thought experiment).
- **“Semi-factual thought experiments**—speculate on the extent to which things might have remained the same, despite there being a different past; and asks the question ‘Even though X happened instead of E, would Y have still occurred?’” (Wikipedia-thought experiment).
- **Prediction** (or forecast)—“attempts to project the circumstances of the present into the future.” (Wikipedia-thought experiment).

- **Hindcasting** (historical re-forecasting)—is “to test (a mathematical) model by observing whether it would have correctly predicted a historical event.” (Collins-hindcast)
- **Retrodiction**—“involves moving backward in time, step-by-step, in as many stages as are considered necessary, from the present into the speculated past to establish the ultimate cause of a specific event.” (Wikipedia-thought experiment).
- **Backcasting**—“involves establishing the description of a very definite and very specific future situation. It then involves an imaginary moving backward in time, step-by-step, in as many stages as are considered necessary, from the future to the present to reveal the mechanism through which that particular specified future could be attained from the present.” (Wikipedia-thought experiment).

Several interesting thought experiments exist in philosophy, science, and education [5, 6].

3.2 Reasons to Use Simulation Experiments

Paul K. Davis

3.2.1 Aren't the Reasons Obvious?

To some, it may seem obvious that we want experiments with models and simulations. Today, however, data analytics is in vogue and it is common to hear questions such as “What does the data say?” as though data speaks to us articulately, precisely, and insightfully. Truth is otherwise. The ideal for science and its applications is a combination of theory-informed experimentation and data analysis on the one hand, and data-informed theoretical development on the other [7, 8]. This chapter discusses some of the reasons.

3.2.2 Relying on Data is Often Impossible

Relying on data is not possible when the needed data is not always available. This may be because

1. The needed data does not exist, although it could reasonably be obtained.
2. The needed data does not exist and could only be obtained with unacceptable delays, expense, or trouble.
3. The needed data cannot be obtained.
4. Data exists, but is unreliable perhaps due to measurement uncertainty, measurement error, non-representativeness, or fraud.

The first two cases need no elaboration. The third, however, is sometimes unappreciated. The most evident example of data that cannot be obtained, outside of

science fiction, is data on the future. Sometimes, of course, historical data can be used as a proxy, as when insurance companies set premiums by extrapolating from historical data and trends. Historical data often does *not* suffice, however, as when one is planning to introduce a new technology, reorganize a company, prepare for possible future wars, or negotiate an international treaty. The changes will create new incentives, new behaviors, and new relationships. The basic problem is that: (1) systems change, sometimes rendering past data obsolete; and (2) even for stable systems, issues arise for circumstances on which past data has not been collected

As for the case of unreliable data, in this era of ubiquitous data connected by the Internet of Everything, it is easy to overlook the fact that the data pouring in does not adhere to experimental standards. It may suffer due, e.g., to measurement uncertainty, measurement error, non-representativeness, and various subtle biases. Also, some of the data may have been deliberately falsified. Historical examples include corruptly invalid battlefield data during the Vietnam War [9], bogus ratings of financial packages before the Great Recession [10], and initial underreporting of COVID-19 cases in China—later acknowledged even by China itself [11].

3.2.3 Even if the Data Exists, It's Not Enough: Understanding and Reasoning Demand Causal Models and, Sometimes, Systematic Experimentation

Even if we have massive, accurate data in a particular domain, and even if accurate predictive empirical models have been developed to answer some questions, we will still often want theory, models and simulation, and related experiments. This will be the case whenever there is need to *understand* the phenomenon at issue (even if imperfectly), or when it is important to *reason* about the phenomenon and its underlying factors and processes. Such understanding and reasoning require *causal* models, and in complicated or complex problems, they require systematic experimentation.

(Causal models are distinct from what are variously called correlations, associations, or statistical models. The issue of causality is deep [12, 13], but it is fundamental to science and the practicalities of life).

The issue of causality is deep, as discussed in several books. A particularly salient discussion is that of Pearl [14], particularly his *Book of Why* [15], written to be accessible to a broad audience of educated people. Policymakers, for example, want to know whether and how to intervene in a system. If their interventions will be more than marginal; i.e., if their interventions will change the system and perhaps the actors within it and/or their incentives, previously collected system data may not even be relevant, much less the basis for estimating consequences.

This is particularly so when dealing with complex or complex adaptive systems [16] something for which simulation experiments are particularly well-suited. This includes using newer methods such as agent-based simulation to generate potential trajectories of complex systems, such as modeled societies [17, 18].

3.2.4 Planning Toward Objectives and Avoiding or Managing Failures

As a first illustration of how simulation experiments can be valuable, consider the history-making *Project Apollo* that led to man reaching the moon and returning safely. Simulations (live, virtual, and constructive), and simulation experiments were used at every stage, as in designing the rocket and landing craft, in planning the trajectory to the moon and the technique for maneuvering its landing craft to the moon's surface, in choreographing the numerous activities undertaken by the crew, such as activities in space and collecting material and data from the surface of the moon itself, and for mission rehearsal.

(Many materials exist describing these applications of simulation, such as those pertaining to engineering simulation [19] or space flight [20])

As a second example, consider the long history of using simulation experiments to avert or manage failures. This function was part of the narrative in Jay Forrester's pioneering work in System Dynamics as he pointed out how intuitively reasonable actions can prove counterproductive because of feedback phenomena, the effects of which become clear with simulation [21]. For example, addressing the problem of urban traffic by building more highways into the city leads over time to new home-building along the new high-ways, which in turn increases the eventual flow into the city [22]. Addressing the problem of urban poverty only with welfare payments and subsidized housing can attract more unemployed people to the area, making things worse, whereas creating attractive areas and effective infrastructure to attract job-providing companies can be more effective (but with its own problems) [22]. Constant economic growth powered by pollution-creating energy sources and avoiding pollution controls can increase GDP until it doesn't, after which the economy may decline disastrously [23]. Despite extreme controversy, the latter work on *Limits to Growth* has proven prescient and held up empirically [24]. The more recent analogue is, of course, the battle over climate change, where climate models have consistently predicted disastrous changes of climate in the absence of radical changes. Again, the model-based work has been controversial but the temperature trends have been accurately predicted [25] and, rather than exaggerating the threat of sea-level rise, modeling has tended to show a bias toward *underestimating* effects—for reasons such as the scientists leaning over backward to avoid being accused of exaggeration [26].

The feasibility of avoiding disaster by using simulation experiments was also championed in Europe by German psychologist Dietrich Dörner under the rubric “Complex Problem Solving” (CPS) using computer-generated scenarios. As with System Dynamics, many of the applications have been to aid executives managing large companies [27].

A recent book by William Rouse discusses a broad range of failure types in business, health systems, and other settings using well-documented cases to illustrate them. Noting that failures will occur, despite best efforts, he argues that computational modeling can help avoid some and manage other failures, in part by establishing mechanisms for recognizing early signs of impending failure [28]. The book builds on previous research on organizational simulation [29] and human-centered decision support [30].

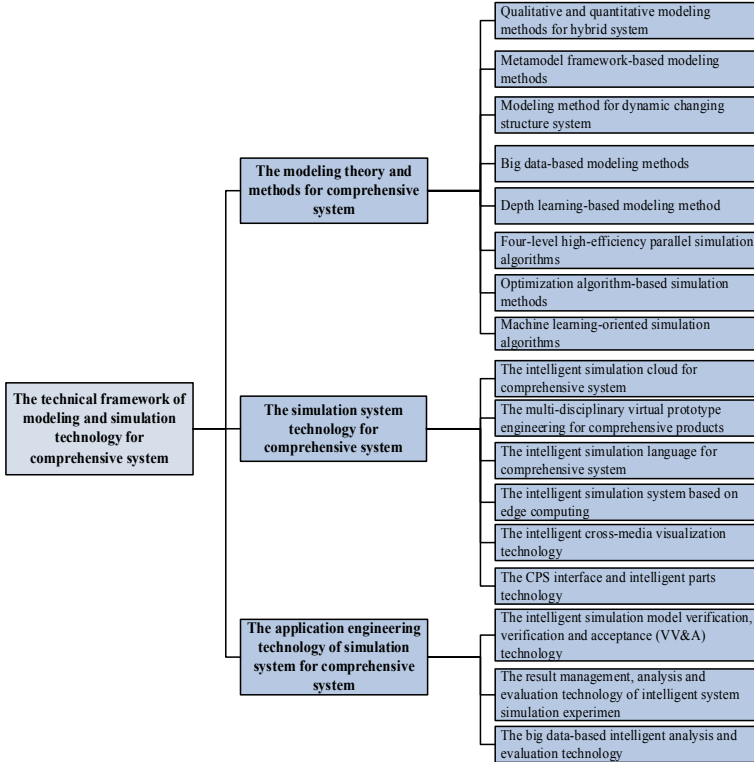
As the last example, during the early months of the COVID-19 pandemic simulation experiments convinced reluctant governments to take extremely consequential decisions in order to avoid millions of deaths beyond those that would otherwise occur. Controversy abounds and time will tell which of the models were most and least sound, but doing nothing was not an option. An early model from Imperial College London influenced decisive actions by the UK and USA [31], and a more empirically driven model from the University of Washington was used continuously in subsequent months [32]. A third effort was notable for being fully public, documented, and posted as an interactive tool allowing state and local officials to conduct simulation experiments on easing restrictions faster or slower [33]. New models on the COVID-19 epidemic were continuing to emerge as this chapter was completed (e.g., the *TRACE* model from the Brookings Institution).

3.2.5 Cautions

3.2.5.1 Model Validity

Unfortunately, models or the data on which they depend are imperfect. It is not just a matter of obtaining the correct input data. Often, on consequential matters, the correct *structure* of the model is uncertain. Does it include all the important variables? Are the cause–effect relationships captured correctly? Does it deal appropriately with random events, both discrete and continual? It has been argued that failure to confront model uncertainty continues to be one of the serious shortcomings in analysis supporting policymakers [34].

Even if the model structure is sound, the model often has many uncertain input parameters, changes of which can have profound effects. This was so with the early models (March to May 2020) of COVID-10 [35]. A key parameter (percentage of infected people reported as infected) was hugely uncertain, leading to predictions of death rate differing by an order of magnitude.



As always, then, it is difficult to exaggerate the importance of assuring the quality of models and their data, as discussed further in Chap. 7 of this volume (Ören et al., forthcoming).

3.2.5.2 Analysis Under Deep Uncertainty

As a related caveat, simulation experiments are frequently beset with the problems of *deep uncertainty* as discussed further in Sect. 7.3.1 and an overview book [36]. In such cases, experimentation should not be about finding an “optimum solution” based on best-estimate assumptions, but rather finding solutions (e.g., a strategy for action) that will prove reasonably good over a wide range of possible assumptions —i.e., across a sizable portion of the parameter-value case space (sometimes called scenario space, case space, input space). The experimentation may then be seen as *exploratory analysis* to support what is called robust decision making (RDM), associated with Lempert et al. [37]. This is drastically different philosophically and practically from merely doing some sensitivity analysis after having identified the nominal optimum solution. See also Chap. 14 (Davis).

3.3 Types of Simulation Techniques for Experimentation

Tuncer Ören

Experimentation is one of the pillars of simulation (Sect. 1.1 Scope). The separation of experimentation and model parts of a simulation study was proposed in a historic document [38]. Specification of the experimental conditions is evolved to experimental frames.

3.3.1 Sections of BoK Guide Related to Experimentation

Due to its importance, several aspects of experimentation are covered in the following sections:

Chapter 1. Preliminary

1.4.2.2 **Experimental** Frame

1.4.2.3 Objectives and **Experimental** Frames

1.4.4.2 **Experimental** Frame—Model Relationships

Chapter 3. Simulation as **Experimentation**

3.1 Types of **Experimentations**

3.2 Reasons to Use Simulation **Experiments**

3.3 Types of Simulation Techniques for **Experimentation**

Chapter 7 Reliability and Quality Assurance of M&S

7.3 Validation

7.3.2.2 Defining Purpose: The **Experimental** Frame

Chapter 16 Philosophy and Modeling and Simulation

16.4 **Experiments** vs Simulation

Chapter 17 History of Simulation

17.2 History of Simulation for **Experimentation** on Digital Computers

3.3.2 Simulation Experimentation for all 3 Types of System Problems

Simulation can be used for all three types of system studies [39, 40], namely for design, analysis, and control problems as outlined in Fig. 3.1.

In **design problems**, the aim is to determine a system which would satisfy a predetermined input/output relationship.

In the simulation of **design problems**, for a design (i.e., a model), the state is given. During simulations with a given model, for input trajectories, output trajectories (or model behavior) are generated. Model behavior, normally, consists of the trajectories of output variables. However, when model structure is variable, the sequence of model structures can also be part of the model behavior. If the model's

System problem	Known			Determine
	Input	State	Output	
Design	Input	State	-	Output
Analysis	Input	-	Output	State
Control	-	State	Output	Input

Fig. 3.1 Use of simulation in system problems

simulated input/output behavior is acceptable, then the model can be implemented as the designed system.

In **analysis problems**, the state of the system is unknown, and the aim is to understand the mechanism of how it functions.

In the simulation of **analysis problems**, input/output behavior of the system is known; the problem is to construct a model (i.e., define state and output variables as well as state transition and output functions) which for some input trajectories, will generate output trajectories comparable to the outputs of the real system under same input trajectories. This would be an iterative process to start with a model (state) and modify the model, until input/output behavior of the model is an acceptable replica of the input/output behavior of the real system.

In **control problems**, a system (hence, its model—with its states, state transition functions, and output functions) is given; the problem is to determine the sequence of the input variable(s), which will cause generation of the output trajectories of the model comparable to the output variables of the real system for the same input variable(s).

The **simulation of control problems**, like the simulation of analysis problems, an iterative process is needed. During the simulation study, one modifies the input trajectories, until a desired output trajectory is obtained.

3.3.3 Relationship of Operations of Simulation and Real System

As outlined in Table 3.3, there are two possibilities so far as the relationship of the operations of simulation and real system are concerned.

3.3.4 Use of Simulation for Decision Support

Simulation experiments are basically used for decision support. Table 3.4 outlines types of simulation used for decision support.

Table 3.3 Possibilities of the relationship of the operations of simulation and real system

Relationship of the operations of simulation and real system	To	Type of simulation	For the purpose of
The system of interest and the simulation program operate simultaneously and provide augmented- (enhanced- or mixed-) reality	Enrich operations of real system	Real-system enriching simulation for	<ul style="list-style-type: none"> – Decision support/online diagnosis – Training – Realistic virtual reality
The system of interest and the simulation program operate alternately to provide predictive displays	Support operations of real system	Real-system support simulation	<ul style="list-style-type: none"> – Decision support – On-the-job training

Table 3.4 Types of simulation used for decision support

Purpose of decision support	Category of simulation	Use of simulation for
Description	Descriptive simulation	
Explanation	Explanatory simulation	
Prediction of behavior/performance	Predictive simulation	
Evaluation of alternative models, parameters, experimental conditions (scenarios), policies	Evaluative simulation for	<ul style="list-style-type: none"> – Feasibility studies – Sensitivity studies – Acquisition (simulation-based acquisition)
Prescription	Prescriptive simulation for	<ul style="list-style-type: none"> – Planning (simulation-based planning) – Online decision support – Engineering design (simulation-based design/simulative design) – Virtual prototyping: (Simulation-based prototyping/simulative prototyping)

3.3.5 Statistical Experiment Design Techniques

Designing simulation experiments is an essential aspect of simulation experiments [41–43]. It has been in practice since the early days of simulation [44].

3.4 Simulation of Discrete Systems, DEVS

Rhys Goldstein, Azam Khan

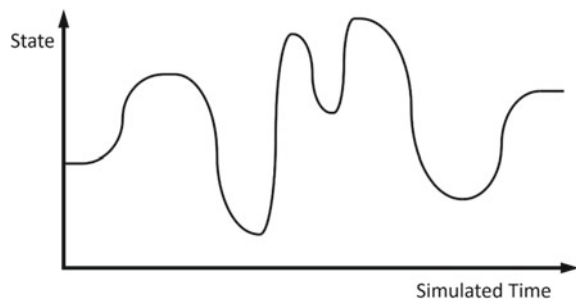
Introduced in 1976, the Discrete Event Systems Specification (DEVS) formalism plays a foundational role in the body of M&S knowledge. DEVS provides a theory-based approach for modeling and simulating systems of essentially any type, with a focus on systems regarded as partially or fully discrete. What distinguishes DEVS from other modeling techniques is the way it incorporates simulated time into the state transition functions that specify the behavior of the represented system. Note that *physical time* is the (DEVS) formalism technical term for “time” as it is perceived in the real world; *simulated time* is a quantitative representation of physical time that is also commonly expressed in units such as years, days, hours, minutes, seconds, and fractions of seconds.

The book *Theory of Modeling and Simulation* by Zeigler et al. [45] serves as the definitive reference on DEVS and other foundational modeling formalisms. Here DEVS is presented from the ground up as special type of state machine that incorporates time into every transition. The presentation begins with a discussion of discrete systems and how they differ from continuous systems.

A *continuous system* is a system in which the state varies continuously over simulated time. An example of a continuous system is illustrated in Fig. 3.2, where the system’s state is plotted on the vertical axis. In general, both continuous and discrete systems can have multi-dimensional states comprising many variables of different types.

Although many real-world systems exhibit smoothly varying quantities like the one depicted in Fig. 3.2, there are also many systems that tend to alternate between long time periods of relative constancy and short time periods of rapid change. For example, an office worker may remain at their desk for an hour or more, then spend less than a minute walking to a conference room or other common area, then spend an hour at the new location interacting with colleagues. If the worker’s position is regarded as the state of the system, then the state will change continuously, but only for the relatively short time periods in which they are walking to a different location. For the long periods of time in which the worker remains in one place, the state will remain constant. Figure 3.3 provides an example of this sort of “nearly discrete” system.

Fig. 3.2 A continuous system. The state varies continuously over simulated time



real-world transition involves a net flow of electrons that increase continuously from zero to some steady-state level. There are countless examples of both artificial and natural real-world systems that can be beneficial to regard as discrete.

DEVS provides a set of conventions for representing discrete systems of more-or-less any domain. Later it will be demonstrated that DEVS can model continuous systems as well. But since DEVS treats state transition as instantaneous, discrete systems are the most obvious application of the formalism.

To understand the rationale for DEVS, it is helpful to consider a more basic and more widely known approach for modeling discrete systems: a state machine. Both state machines and DEVS feature discrete events, or *events* for short, which are self-contained sets of operations that occur at the points in time at which a discrete system may transition from one state to another. The key difference between a state machine and DEVS is that DEVS incorporates simulated time into these transitions.

State machines are based solely on *virtual time*, a time representation that orders events in a way that is consistent with causality and consistent with simulated time. The *consistent with causality* (causal consistency) property tells us that if one event influences a second event, then the second event occurs at a later point in virtual time. The *consistent with simulated time* property means that if one event occurs before a second event in simulated time, then the second event occurs at a later point in virtual time as well. What is not captured by virtual time is the real-world duration of time between events. Two successive events in virtual time may be separated by any duration of simulated time, including zero duration. Figure 3.5 depicts the same system as illustrated in the previous figures, but with the state plotted over virtual time. Each segment of the state trajectory is shown as the same width, even though the simulated time durations vary.

Because a state machine is based on virtual time only, its events may not be triggered by the passage of simulated time. Instead, each event must be triggered by an input. One approach to formulating a state machine is to associate each event with an output as well. In Fig. 3.6, the same system that appeared in previous figures is shown with states labeled s_1 through s_7 . Each transition from one state to

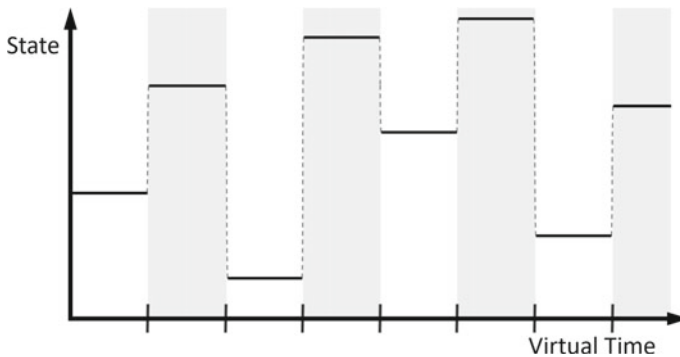


Fig. 3.5 A discrete system plotted over virtual time. The real-world duration associated with each state is not represented

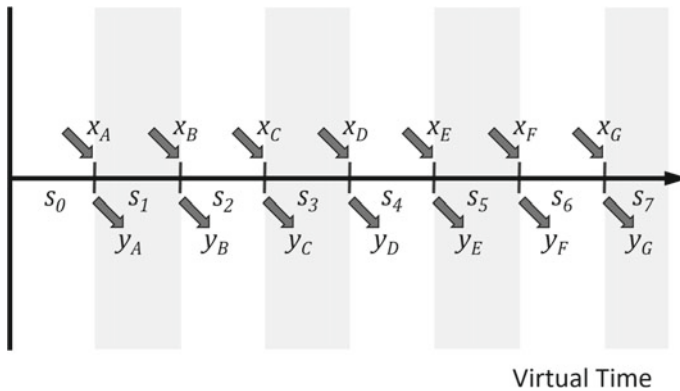


Fig. 3.6 A discrete system represented by a state machine. Each event is triggered by a single input and, in this example, produces a single output

the next is triggered by an input labeled x_A through x_G and produces an output labeled y_A through y_G .

A state machine can be defined using a function f that is invoked at each event in response to an input. As shown below, the function may take the current state s and the input x as arguments, and produce the new state s' and the output y .

$$s', y = f(s, x)$$

Although they serve a foundational role in the specification of computing systems, state machines are limited in that they do not inherently represent the durations of time associated with real-world processes. DEVS addresses this limitation by incorporating simulated time into the state transitions that govern the behavior of a discrete system. In essence, a DEVS model is a state machine with time included as a component of the state.

In the DEVS literature, the term *state* still refers to the set of variables that remain constant between events. However, a new variable called the *elapsed duration* is introduced to represent the continuously increasing duration of simulated time that has elapsed since the previous event. As shown in Fig. 3.7, the elapsed duration falls instantaneously to zero whenever the state transitions, then increases linearly until the next event.

Together, the state and the elapsed duration are referred to as the *total state*. The normal convention is that the state is denoted s , and here the elapsed duration is denoted Δt_e . The total state is therefore $[s, \Delta t_e]$. A DEVS model can be regarded as a state machine that specifies transitions of the total state $[s, \Delta t_e]$ —that is, transitions of the state and elapsed duration in combination—rather than transitions of the state s by itself.

Combining the state and elapsed duration introduces a need for two types of state transitions. One type of state transition is analogous to that of a state machine in that it is triggered by an input from an external source. These externally triggered state

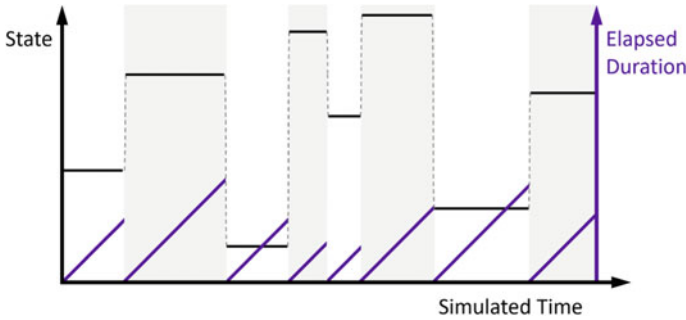


Fig. 3.7 A discrete system represented by the state in combination with the elapsed duration

transitions, or *external transitions* for short, are specified by an external transition function δ_{ext} . In essence, the external transition function transforms the current total state $[s, \Delta t_e]$ into a new total state $[s', \Delta t_e']$ using the input value x as an additional source of information. However, the new elapsed duration $\Delta t_e'$ is always zero. The result of δ_{ext} is therefore just the new state s' by itself, as shown below.

$$s' = \delta_{\text{ext}}([s, \Delta t_e], x)$$

The other type of transition is triggered by the total state itself. Consider that the elapsed duration Δt_e is a continuously increasing quantity. It follows that the total state $[s, \Delta t_e]$ also varies continuously with time and may in fact reach a point at which it is no longer desirable for the system to remain on its current trajectory. If such a point is reached, a transition occurs. These internally triggered state transitions, or *internal transitions*, are specified by the internal transition function δ_{int} . In essence, the internal transition function transforms the current total state $[s, \Delta t_e]$ into a new total state $[s', \Delta t_e']$. But as with the external transition, the new elapsed duration $\Delta t_e'$ is always zero, so the result of δ_{int} is just the new state s' . Also, as explained below, the current elapsed duration Δt_e can also be omitted because it has a known value. Hence, as shown below, the definition of the internal transition function includes the current and new states but excludes any explicit quantity of time.

$$s' = \delta_{\text{int}}(s)$$

The internal transition occurs at the point where the continuously increasing elapsed duration Δt_e reaches a certain duration that depends on the state. That duration is specified by the *time advance* function ta , which is a function of s . The reason why the elapsed duration is not an argument of δ_{int} is because, at that moment of the simulation, the elapsed duration is known to be $ta(s)$.

$$\Delta t_e = ta(s) \quad (\text{at internal transitions})$$

In DEVS, outputs are associated with internal transitions and not external transitions. This convention is a practical one, as it allows there to be an arbitrary duration of time separating an input from the subsequent output. If $ta(s) = 0$, then an output may occur immediately after an input with no advancement of simulated time in between. But if $ta(s) > 0$, an output can be arranged to occur after an arbitrarily long delay. The output value is specified by the output function λ , which is a function of the state prior to the invocation of δ_{int} . In essence, the output function could be considered a function of the total state $[s, \Delta t_e]$, but Δt_e is not needed because it is known that $\Delta t_e = ta(s)$ at that moment of the simulation.

$$y = \lambda(s)$$

In the same way that a lone function f can specify the behavior of a state machine, the four functions δ_{ext} , δ_{int} , λ , and ta specify the behavior of a DEVS model. To complete a DEVS model specification M , one defines all four functions as well as the set of all possible input values X , the set of all possible output values Y , and the set of all possible states S .

$$M = \langle X, Y, S, \delta_{\text{ext}}, \delta_{\text{int}}, \lambda, ta \rangle$$

For an example of a DEVS model specification, consider simulating a person in a workplace. The worker exhibits the following behavior:

1. At any point in time, the worker is either working or taking a break.
2. When the worker begins working, they continue working for 1 h before starting their break.
3. When the worker begins taking a break, they initially intend to resume working in 10 min.
4. Despite their initial intent, the actual duration of the worker's break is influenced by their co-workers. Specifically, if a co-worker transitions from working to taking a break, then the original worker increases the length of their break. In such cases, they remain on their break for a duration of $(\Delta t_p^2 - \Delta t_e^2)^{1/2}$, where Δt_p is the duration they were planning to remain on break at the previous transition (which occurred at a duration of Δt_e in the past). If a co-worker on break resumes working, then the original worker decreases the length of their break. They resume working themselves after a duration of $\Delta t_p' = \Delta t_p - (2 \cdot \Delta t_p \cdot \Delta t_e - \Delta t_e^2)^{1/2}$.

The example aims to capture a social behavior that may occur in certain work environments. If there are n workers taking a break and an $(n + 1)$ th worker joins them, they are compelled to interact with their colleague and thus extend the remainder of their breaks. But if there are n workers taking a break and one of them leaves to resume working, the remaining $n - 1$ workers are reminded of the action of leaving and consequently shorten the remainder of their breaks.

The behavior of a worker in this scenario is described by the DEVS model M_{worker} .

$$M_{\text{worker}} = \langle X, Y, S, \delta_{\text{ext}}, \delta_{\text{int}}, \lambda, ta \rangle$$

DEVS model inputs and outputs are often associated with ports. In this case, there is a single input port that receives the change in the number of other workers taking a break. As indicated by X , these inputs take the form of the port ID “ Δn_{break} ” followed by the received value Δn_{break} .

$$X = \{ [\text{"}\Delta n_{\text{break}}\text{"}, \Delta n_{\text{break}}] \mid \Delta n_{\text{break}} \in \mathbb{Z} \}$$

There is also one output port, named “task,” that sends out the task that has just begun. As specified by Y , the sent value $task$ is either “work” or “break.”

$$Y = \{ [\text{"task"}, task] \mid task \in \{ \text{"work"}, \text{"break"} \} \}$$

The state consists of two variables: the task currently being performed ($task$) and the duration of time after which the worker is planning to switch tasks (Δt_p). The definition of S below indicates that $task$ is either “work” or “break,” and the planned duration Δt_p is either zero or any positive real number.

$$S = \{ [task, \Delta t_p] \mid task \in \{ \text{"work"}, \text{"break"} \}, \Delta t_p \in \mathbb{R}_0^+ \}$$

When an input is received, an instance of the DEVS model responds differently depending on whether it is in the “work” or “break” state. If $task = \text{"work"}$, then the input is essentially ignored. To ignore the input, it is necessary to update the planned duration Δt_p by subtracting the elapsed duration Δt_e . The new planned duration $\Delta t'_p = \Delta t_p - \Delta t_e$ effectively schedules the next internal transition at the point in simulated time that it would have occurred had there been no input at all. If the input value Δn_{break} is received when $task = \text{"break"}$, then the length of the break is adjusted according to the calculations specified in the description of the model. The worker’s response to an input is formally defined by the external transition function δ_{ext} .

$$\begin{aligned} \delta_{\text{ext}}([[task, \Delta t_p], \Delta t_e], [\text{"}\Delta n_{\text{break}}\text{"}, \Delta n_{\text{break}}]) &= [task, \Delta t'_p] \\ (\text{task} = \text{"work"}) &\Rightarrow (\Delta t'_p = \Delta t_p - \Delta t_e) \\ (\text{task} = \text{"break"}) &\Rightarrow \dots \\ (\Delta n_{\text{break}} = 0) &\Rightarrow (\Delta t'_p = \Delta t_p - \Delta t_e) \\ (\Delta n_{\text{break}} > 0) &\Rightarrow (\Delta t'_p = (\Delta t_p^2 - \Delta t_e^2)^{1/2}) \\ (\Delta n_{\text{break}} < 0) &\Rightarrow (\Delta t'_p = \Delta t_p - (2 \cdot \Delta t_p \cdot \Delta t_e - \Delta t_e^2)^{1/2}) \end{aligned}$$

When the planned duration (Δt_p) elapses, an internal transition must be triggered so that the worker can transition from working to taking a break, or vice versa. The state variable *task* changes from “work” to “break” or “break” to “work.” The planned duration is also updated with the amount of time the worker intends to perform the new task, either 3600 s (1 h) if they are now working or 600 s (10 min) if they are now taking a break. These planned state changes are formally defined by the internal transition function δ_{int} .

$$\begin{aligned} \delta_{\text{int}}([\text{task}, \Delta t_p]) &= [\text{task}', \Delta t_p'] \\ (\text{task} = \text{"work"}) &\Rightarrow \dots \\ \text{task}' &= \text{"break"} \\ \Delta t_p' &= 600 \\ (\text{task} = \text{"break"}) &\Rightarrow \dots \\ \text{task}' &= \text{"work"} \\ \Delta t_p' &= 3600 \end{aligned}$$

Immediately before the internal state transition occurs, the output function λ is invoked to provide the output value. The value is simply the worker’s new task, which is the opposite of the current task.

$$\begin{aligned} \lambda([\text{task}, \Delta t_p]) &= [\text{"task"}, \text{task}'] \\ (\text{task} = \text{"work"}) &\Rightarrow (\text{task}' = \text{"break"}) \\ (\text{task} = \text{"break"}) &\Rightarrow (\text{task}' = \text{"work"}) \end{aligned}$$

The time advance *ta* function expresses the fact that, provided there are no intervening inputs, the internal transition will occur when the elapsed duration reaches the value of the state variable Δt_p .

$$ta([\text{task}, \Delta t_p]) = \Delta t_p$$

Figure 3.8 shows the progression of a simulation with 12 workers who adjust their break schedule in response to their co-workers. The 12 schedules are initially staggered by 5 min, but eventually the workers separate into two groups. In each group, workers start and end their breaks at the same time.

The worker model is just one example of a discrete system that can be modeled using the DEVS formalism. By incorporating time into every state transition, DEVS can be used to represent nearly any time-varying system. Due to its generality, DEVS can be considered foundational to modeling and simulation in much the same way that state machines are foundational to conventional software development.

An important practical feature of DEVS is the ability to couple DEVS models so they can interact with one another. These interactions take the form of messages that originate from the output of one model instance and are then treated as inputs

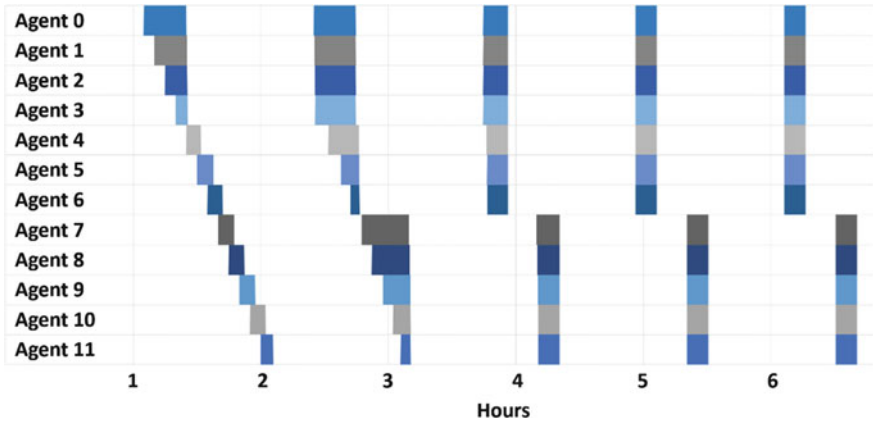


Fig. 3.8 A timeline produced by 12 instances of the worker example model. The workers enter the system at 5-min intervals starting in the “work” state. The colored bars indicate when each worker is taking a break. Although the breaks are initially staggered, a behavior emerges in which the breaks become synchronized within two distinct groups of workers

for other model instances. If an output occurs at a certain point in simulated time, then it is received at the same time point. (In virtual time, however, a message is received at a later point than when it is sent.)

Couplings between DEVS model instances are defined in the context of a *coupled model*, which is itself a form of DEVS model. A coupled (or “network”) model N is specified by supplying eight elements. Two of these elements—the input set X and the output set Y —are the same as in the basic specification.

$$N = \langle X, Y, D, \{M_d | d \in D\}, EIC, EOC, IC, Select \rangle$$

The elements of a DEVS coupled model specify a graph-like or network-like structure involving instances of DEVS models referred to as *components*. Using the coupled model in Fig. 3.9 as an example, the component name set D contains the IDs of the three components (“A,” “B,” and “C”), and the component set $\{M_d\}$ contains the corresponding DEVS model specifications ($\langle X_A, Y_A, \dots \rangle$, $\langle X_B, Y_B, \dots \rangle$, $\langle X_C, Y_C, \dots \rangle$). The external input coupling EIC formally represents all the purple links (arrows) in the diagram, which direct messages from the inputs of the overall coupled model to the inputs of the components. The external output coupling EOC represents the green links, which direct messages from the outputs of the components to the outputs of the coupled model. The internal coupling IC represents the blue links, which direct messages from the outputs of some components to the inputs of other components.

During a simulation, it may happen that multiple components are scheduled to undergo an internal state transition at the same point in simulated time. In such cases, the *Select* function is invoked to identify the component that should transition first. The *Select* function is part of the original formulation of the theory,

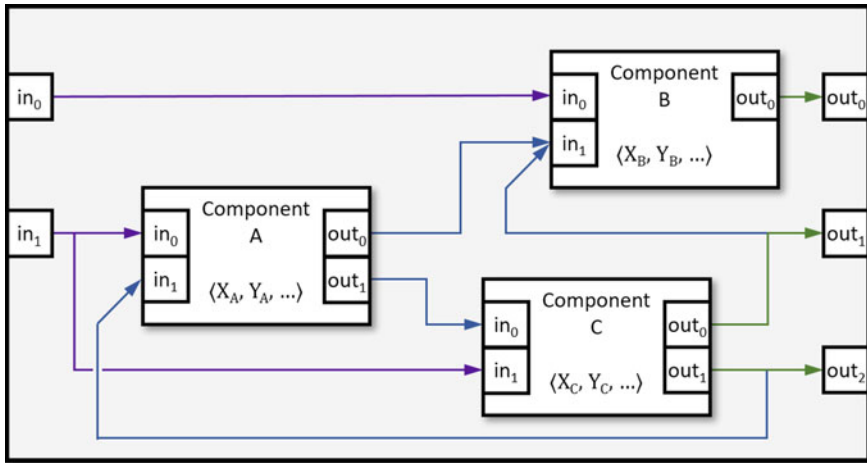


Fig. 3.9 An example of DEVS coupled model

known as *Classic DEVS*. It is worth noting that a variant of DEVS called *Parallel DEVS* was introduced in 1994 to eliminate the Select function and allow the internal transitions of multiple components to be invoked in a synchronous fashion. There are many other subclasses and variants of DEVS, including Cell DEVS, Dynamic Structure DEVS, Multi-Level DEVS, Routed DEVS, and Symmetric DEVS, to name a few. Most of these derived formalisms are based on either Classic DEVS or Parallel DEVS.

A component of a DEVS coupled model may itself be a coupled model. This feature is enabled by a property of DEVS models known as *closure under coupling*, which means that any coupled model specification can be mapped onto a basic DEVS model specification with external and internal state transitions. It is impossible to tell, for example, whether each of the three components in Fig. 3.9 is an atomic or coupled model. Note that if Component A is a coupled model, then some of its components could also be coupled models. The ability to nest coupled models allows complex systems to be defined as hierarchies of simpler systems.

Although the primary focus of DEVS is the representation of discrete systems, the formalism can be used to model continuous systems as well. Again, the key insight is that DEVS is essentially a state machine that specifies transitions of the total state: the state s in combination with the elapsed duration Δt_e . Whereas s is a discrete function of time, the total state varies continuously due to the linear increase of Δt_e between transitions. The downside to representing continuous systems using DEVS, however, is that either an analytic or a numerical solution to one or more differential equations must generally be incorporated into the model to derive the continuously varying quantities of interest from the total state.

To demonstrate the representation of continuous systems using DEVS, consider the following differential equations expressing the height h and velocity v of a falling object subject to a gravitational acceleration g .

$$\begin{aligned} dh/dt &= v \\ dv/dt &= -g \end{aligned}$$

A DEVS model can track the continuous quantities h and v by integrating the differential equations from the time of the previous event to the current time, a duration that is always equal to the elapsed duration Δt_e . The updated quantities h' and v' have analytic solutions, which are given below for external transitions where Δt_e is available as a function argument. For internal transitions, one can replace Δt_e with $ta(s)$.

$$\begin{aligned} h' &= h + v \cdot \Delta t_e - (1/2) \cdot g \cdot \Delta t_e^2 \\ v' &= v - g \cdot \Delta t_e \end{aligned}$$

Given a set of differential equations with no analytic solution, a numerical solver would have to be incorporated into the DEVS model to produce the trajectories of the continuous values. Note that in sophisticated models, the trajectories of continuous quantities are likely to change whenever certain inputs are received.

Regardless of the system under investigation, the DEVS formalism can be applied in several ways. Modelers seeking to take full advantage of the formalism may create formal system specifications using mathematical conventions like those presented above. It is also possible to implement DEVS models without first specifying them. DEVS models are generally implemented using M&S libraries developed for mainstream procedural and object-oriented programming languages such as Python, Java, or C++. A third option is to implement a simulation model using a different paradigm or set of conventions and automatically map the authored model onto an equivalent DEVS model. With this third approach, modelers can use domain-specific languages dedicated to their areas of expertise, then couple the resulting models as if they had been implemented using DEVS conventions. In all three approaches, a DEVS-based simulation framework serves as a platform supporting the integration of system models from any number of application domains.

3.5 Simulation of Continuous Systems

Amine Hamri, Laurent Capocchi

The simulation of dynamic systems and especially of continuous systems allows reproducing the expected behavior of such systems on a computer (calculator) for verification and validation. However, the lack of concepts to model and simulate directly continuous systems forces the modeler to discretize some variables: the time, state variables, or both of them. Recently, many scientists have proposed paradigms and techniques to discretize such variables. In opposite to time discretization modeling and simulation techniques that are widely used by different communities, a small group of scientists at his head the professor Bernard P. Zeigler

has highlighted the discrete event simulation techniques that provide a clear framework for the modelers. We can cite Discrete Time Systems Specification (DTSS), Quantized DEVS (Q-DEVS), Generalized DEVS (G-DEVS) that allow modeling and simulation of dynamic systems more faithfully with some advantages and disadvantages [45].

For example, G-DEVS [46] models the state trajectory of dynamic system with piece-wise polynomial functions instead of piece-wise constants like in DEVS. This characteristic enhances the simulation results of many applications like digital circuits and moving agents. The lack of such formalisms is that suffering from the powerlessness of formal and analytical techniques like theorem-proving to give solutions. However, these formalisms-oriented simulation remain excellent tools to understand and analyze continuous systems where analytical techniques fail due to the complexity of dynamic systems that increases more and more.

3.6 Hybrid Modelling and Simulation

Navonil Mustafee, Mamadou Kaba Traoré

We begin with an introduction to the idea of hybridization, both within the modeling and simulation (M&S) discipline itself as well as between M&S and other disciplines, and follow this with a formalization of hybrid M&S.

3.6.1 From Complexity to Hybridization

Managing complex systems can be tedious [47] not only because of a huge number of subcomponents that may compose them but also because of the complex processes that govern the relations that exist between them rendering their analysis and design more difficult. Modern complex systems require multiple levels of explanation to be provided to achieve their various objectives, while keeping a holistic understanding of the behavioral pattern of the overall system and its interaction with the surrounding environment [48]. As such, a hybridization of approaches that would evidently provide useful knowledge from various angles on how such systems perform at the holistic level rather than focusing on specific problems in isolation for specific solutions is an appropriate means to address their complexity. In modeling and simulation (M&S), such a hybridization can be envisioned endogenously or exogenously, and at different levels of concern [49–51].

As described in Table 3.5, the concepts level, where the universe of discourse is set (such as the notions of state, event, and concurrency), calls for formalisms and (more generally) methods to capture the required concepts in a symbolically manipulable way. While the M&S community traditionally distinguishes between discrete and continuous phenomena as regard to central time-related concepts, qualitative and quantitative computational approaches, such as operation research

or artificial intelligence methods, rather focus on problem-solving steps and mechanisms. Hybridization comes at this level with the objective-driven need to deal with temporal considerations for the system under study while trying to find a solution to the problem under study. Such a situation happens for example when optimization techniques make use of simulation as a black box-type of evaluation function (exogenous hybridization), or when the requirement for a fine-grained understanding of the system entails both continuous and discrete phenomena be considered (endogenous hybridization). At the specification level, the real-world system and problem under study are expressed as a model, using the universe of concepts adopted, i.e., discrete or continuous simulation model (within M&S world) or problem-solving algorithm (within the wider computational world). The literature has coined various terms to qualify the various possible hybridizations, such as DisM + ContM, or DisM/ContM + Alg (where “+” denotes a composition/mixing operation that can vary from loose to tight integration). At the operations level, engines are built to execute the model defined at the immediate upper level. Such engines are often referred to in the M&S world as simulators and integrators (for respectively discrete and continuous operations), while solvers implement the algorithms defined in non-M&S-centered computational approaches. Operational hybridization occurs here to support the requirement for multiple execution engines, each devoted to aspects that other engines do not support. The hybridization done at the operations level between computational engines and physical components is the essence of the so-called Cyber-Physical Systems. The hierarchy of levels in Table 3.5 implies stronger hybridization at the upper level and weaker hybridization at the lower level.

Table 3.5 Hybridization strategies in computational frameworks

<i>Concepts (formalisms)</i>	DEVS, Petri Net, Multi-Agents...	ODE, PDE, System Dynamics...	OR methods, AI methods...	
<i>Specifications (models)</i>	Discrete simulation models (DisM)	Continuous simulation models (ContM)	Algorithms (Alg)	
<i>Operations (engines)</i>	Simulators (Sim)	Integrators (Int)	Solvers (Sol)	Physical devices (Phy)
	M&S world (SimW)			
	Computational world (CompW)			

- DEVS: Discrete Event System Specification
- PDE: Partial Differential Equations
- AI: Artificial Intelligence
- : often referred to as combined simulation
- ODE: Ordinary Differential Equations
- OR: Operation Research
- : often referred to as hybrid simulation
- : CPS

3.6.2 Definitions

Hybrid Simulation is the combined application of simulation techniques like System Dynamics (SD), Discrete Event Simulation (DES), and agent-based Simulation (ABS) in the context of a single simulation study. Its objective is to better represent the system under scrutiny through the conjoined application of multiple simulation techniques. Traditionally, modelling efforts have primarily been undertaken in distinct M&S communities, each with their International Societies, Conferences, and Journals. For example, the SD, DES, and ABS communities have continued to thrive under scholarly societies like the System Dynamics Society, The Society for Modeling and Simulation International, and European Social Simulation Association; each community has its conferences, e.g., International Conference of the System Dynamics Society, Winter Simulation Conference, Social Simulation Conference, and scholarly publication outlets, e.g., System Dynamics Review, Simulation: Transactions of the SCS, the Journal of Artificial Societies and Social Simulation. Hybrid Simulation presents the opportunity to draw on these different world views and structured methods of system representation and to realize synergies through mixing methods and applying them to model the increasingly complex systems of today.

Hybrid Modelling is the combined application of simulation with methods and techniques from disciplines such as Computer Science/Applied Computing, Operations Research (OR), Humanities, Engineering, and Data Science. Unlike Hybrid Simulation, whose focus continues to be inward to the M&S community, Hybrid Modelling proposes a cross-disciplinary approach for the best possible representation of a system, more specifically, the combined application of simulation, or indeed Hybrid Simulation, with methods, techniques from broader disciplines. Examples include the use of game-theoretic approaches (Economics) with computer simulation, faster execution of simulations using Grid, Cloud, GPGPU and Parallel Computing technologies (Computer Science), formal testing (Software Engineering) of simulation models, use of problem structuring methods and Qualitative System Dynamics (Soft OR) in the problem formulation and conceptual modelling stages of a simulation study, the combined application of load plan heuristics (Hard OR) with computer simulation. As illustrated in Fig. 3.10, Hybrid Modelling is cross-disciplinary and, at its very core, is a call to the M&S community to engage with researchers from diverse disciplines, and learn the extant knowledge and underlying philosophies and methodologies, frameworks and techniques (subsequently referred to as discipline-specific artifacts) with the aim of extending the theory and practice of M&S. In the context of Hybrid Modelling, it is important to note that the discipline-specific artifacts could be used in one or more stages of a simulation study, for example, conceptual modelling phase, model coding, input data analysis, V&V, scenario development, experimentation and implementation of the results of a simulation study. Mustafee and Katsaliaki [52] identified a plethora of OR methods and techniques, some of which are already used together with simulation approaches (e.g., the combined application of forecasting with DES—Harper et al. [53], Harper and Mustafee [54]). Yet, others are avenues for future research.

3.6.3 The Unified Conceptual Representation of Hybrid Modelling and Hybrid Simulation

Mustafee and Powell [56] present a unifying conceptual representation of Hybrid Simulation and Hybrid Modelling to clarify the terminologies further and put them in perspective. With increasing interest in Hybrid Simulation and debates around its definition, scope, and purpose [57], it was felt that it was essential to present the definition of Hybrid Modelling, which was distinct from Hybrid Simulation, and without which the community world continue to look inwards (Fig. 3.10). In developing the definition of Hybrid Modelling and Simulation, the authors considered its alignment with the historic (albeit infrequent) use of the term—see [51]—the past (Case 1); the use of Hybrid Simulation in present-day academic discourse—the present (Case 2); and, the use of the term to support future research in advancing the theory and practice of M&S—the future (Case 3). Figure 3.11 presents the classification of Hybrid Modelling and Simulation into distinct Model Types. In laying the basis of the unified definition, they revisited [58] definition of paradigm, methodology, technique, and tool and adapt it for hybrid study.

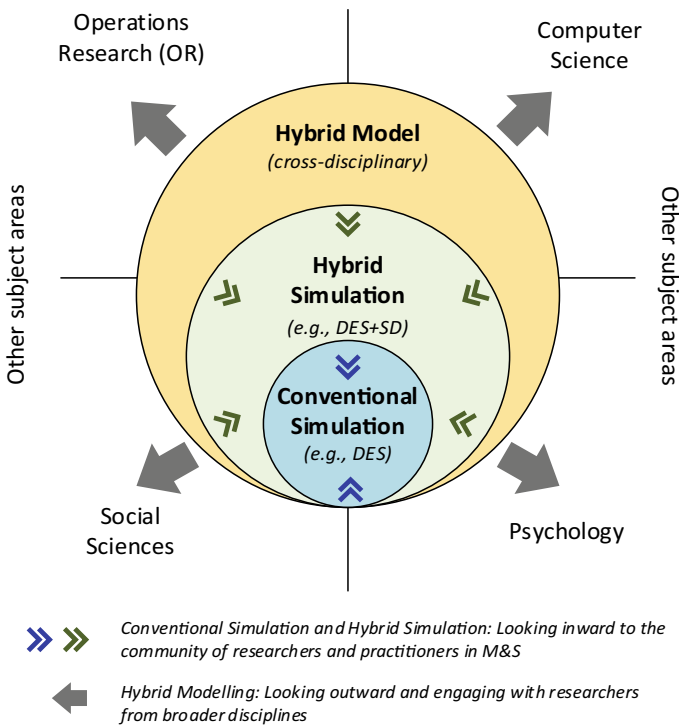


Fig. 3.10 Hybrid models focus on cross-disciplinary engagement (adapted from Fishwick and Mustafee [55])

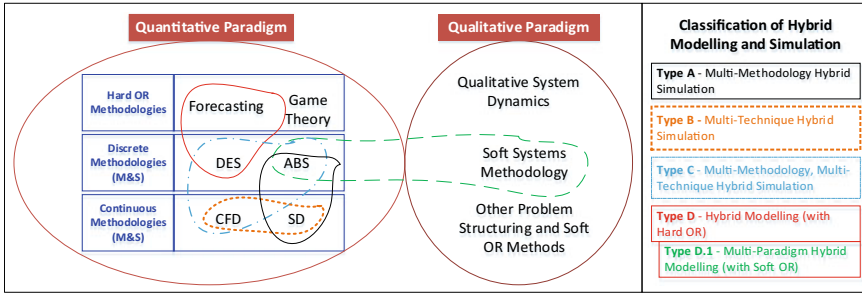


Fig. 3.11 Unified conceptual representation of hybrid modelling and simulation into distinct model types (adapted from Mustafee and Powell [56])

- **Paradigm:** They distinguish between qualitative (interpretive, subjective, soft) and quantitative (positivist, objective, hard) paradigms; M&S is in the quantitative paradigm. If qualitative approaches are used, e.g., in conceptual modelling phase, then it is an example of Multi-Paradigm Hybrid Modelling.
- **Methodology:** Methodologies develop within a paradigm and usually embody its philosophical assumptions (ibid.). In the quantitative paradigm, [59] distinguish between discrete and continuous methodologies. In discrete execution of computer models, the system state changes from one event to the next (as in DES) or as per defined time steps (as can happen in both DEA and ABS). For continuous simulation, the change in system state is continuous (as with SD and Computational Fluid Dynamics, or CFD). A Multi-Methodology Hybrid Simulation is one which has both Discrete and Continuous elements, e.g., SD-DES, SD-ABS.
- **Technique:** Techniques exist within the context of methodologies and have well-defined purposes, e.g., DES (ibid). Mustafee and Powell [56] distinguish between techniques such as DES (event list/queuing theory) and ABS (time stepped/emergence) under discrete methodology, and SD (stock and flow) and CFD (numerical approach) under continuous methods. A Multi-Technique Hybrid Study is one which uses two or more techniques under the same methodology, e.g., using CFD to model traffic flow with SD to investigate strategic policy related to transportation at an urban level. It follows that a Multi-Methodology, Multi-Technique Hybrid Simulation is one which uses a combination of techniques from both discrete and continuous methodologies, with at least two techniques from either of the two methods. Studies demonstrating the combined application of SD-DES-ABS are an example of this.
- **Tool:** We define these as M&S packages which can be used to “perform a particular technique” (ibid.), and more recently, can execute multiple techniques that are classified under one or more methodologies. Discussion of the tool is not important for the hybrid modelling and simulation classification scheme.

3.6.4 Classification of Hybrid Modelling and Simulation into Distinct Model Types

- Type A—Multi-Methodology Hybrid Simulation—Models of this type align with present practice (Case 2). Numerous studies used SD-DES and SD-ABS.
- Type B—Multi-Technique Hybrid Simulation—Although these align with present practice (Case 2, e.g., use of ABS-DES models), there is some debate as to whether these could be called as hybrid since both techniques conform to discrete methodologies. In [56] classification, a combined application of ABS-DES is Type B Hybrid Simulation since there are fundamental differences in the execution of the simulation logic, which makes them agreeable to model particular category of problems (top-down queuing approach versus bottom-up emergence).
- Type C—Multi-Methodology, Multi-Technique Hybrid Simulation—This aligns with Case 2 (present practice, e.g., ABS-DES-SD models) and also accommodates future hybrid studies (Case 3).
- Type D—Hybrid Modelling—This aligns with Case 1 and encompasses [51] original use of Hybrid Simulation/Analytical Model and the four defined Classes of such models. An example of Type D model is the combined application of mathematical modelling/optimization approaches with simulation models, e.g., use of load plan heuristics with ABS [60]. However, Hybrid Modelling is also Case 3 (the future), since there are numerous well-defined methods and techniques to problem-solving in Operations Research—refer to the classification of OR [52]—and which could potentially be used in one or more stages of a simulation study. Also, as shown in Fig. 3.12, OR is but one of the many disciplines for cross-disciplinary collaboration with M&S (Case 3).
- Type D.1—Multi-Paradigm Hybrid Modelling—When Soft OR techniques are used with M&S, e.g., Soft Systems Methodology and Qualitative System Dynamics, then we have a special case of Type D model which interests paradigm. This is our Type D.1 or Multi-Paradigm Hybrid Model.

Types D and D.1 are referred to as Hybrid Model (rather than Hybrid Simulation) since only one constituent of the combined model is a simulation model; the other component is a discipline-specific artefact (which could be a philosophy,

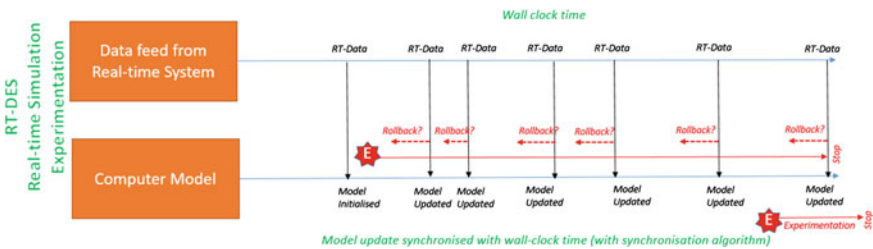


Fig. 3.12 RT-DES real-time simulation experimentation using synchronization algorithm

methodology, framework, technique or a structured approach) that could be combined in one or more stages of an M&S study. For Type A, Type B, and Type C Hybrid Simulation, please refer to the literature review paper “Hybrid Simulation Modelling in Operational Research: A State-of-the-art Review” [61]. Refer to Mustafee and Bischoff (2016) for an example of Type D Hybrid Model. Refer to Powell and Mustafee [62] for a review of literature on Type D.1 models.

3.7 Real-Time Discrete-Event Simulation

Alison Harper, Navonil Mustafee

3.7.1 Overview:

Real-time discrete-event simulation (RT-DES) uses real-time or near real-time data in a computer simulation. RT data can inform different stages of a simulation study. For example, in the conceptual modeling phase, RT data streams can inform the scope and the modeling objectives. In the input data analysis stage, the use of RT data enables us to recompute the distributions more frequently than has traditionally been possible (most simulations rely on distributions derived from historical data). In the model implementation stage, RT data can populate key variables at run-time, for example, the length of the queues, the number of servers available (e.g., machine breakdown would affect the replication count), and updated server processing time (computed real-time). In the model validation stage, the output of the simulation can be compared with the real system; the comparison is especially valid for simulations developed for short-term decision making.

With the growth of Industry 4.0 and the widespread use of ubiquitous computing technologies such as Internet of Things (IoT), RT data feeds from sensors and enterprise information systems are more readily available for subsequent processing. RT data can be used in an overarching data analytics framework comprising descriptive, predictive, and prescriptive approaches. An example of this is the RH-RT data analytics framework for reducing wait time at hospitals [63]. **Descriptive analytics** (including business intelligence) has been the primary consumer of RT data, for example, updating the location of a fleet of cars using GPS information being relayed in real time. **Predictive analytics** has also used RT data (as time series) for short-term decision making, for example, by applying forecasting algorithms on RT data streams [64, 65]. The focus of RT-DES is **prescriptive analytics**. Here, RT data is used for both the modeling process (conceptual modeling, input data analysis, model logic, validation and verification, scenario development) and its eventual execution (updating distributions, global and local variables, and flow control). Arguably, the use of RT data is more challenging for the latter (i.e., real-time experimentation), as it may require

synchronization with time-stamped data streams. This is the realm of parallel and distributed simulation and the use of synchronization protocols [66].

The focus of this technical note is on real-time simulation experimentation. In Sect. 3.2, we identify two forms of RT-DES and provide illustrations that conceptualize the execution of two forms of real-time models and have identified key literature.

3.7.2 Two Forms of RT-DES

RT-DES using time-stamped data may need to incorporate synchronization algorithms to prevent causality errors. The concept of digital twins arguably adheres to this **real-time simulation experimentation** paradigm where RT data continually updates the DES model (Fig. 3.12). In such cases, the simulation time and wall-clock time can be kept *in sync* using conservative and optimistic synchronization protocols. For example, Fig. 3.12 illustrates a scenario where the simulation time is ahead of the wall-clock time. However, upon receiving time-stamped data from the real-time system, the simulation may be rolled back to a previous state. The rollback is necessary to prevent causality errors. Examples of synchronization algorithms include Chandy–Misra–Byrant conservative time synchronization [67, 68] and Jefferson’s Time Warp optimistic algorithm [69].

Faster than real-time experimentation is our second form of RT-DES. This experimentation strategy is implemented when the model automatically executes the pre-defined scenarios upon receiving a tranche of RT data (no further data updates are allowed until the simulations have concluded). The subsequent set of experiments can then be executed when new RT data is received. Note that the newly received data will often be a subset of the overall data being used by the model. As such, there may be logic to determine whether the new feed of RT data necessitates a fresh execution of the scenarios. In Fig. 3.13, it is assumed that data is received every half-an-hour and is significant. Thus, the model is initialized every thirty minutes and the experiments are executed until the simulation end time (shown as stop). The insights gained from the results of such faster than real-time experiments may enable decision-makers to swiftly make changes to the real-world system, with the intended objective of preventing bottlenecks that may have been

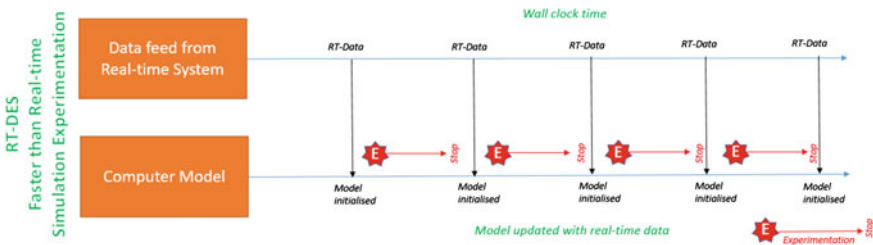


Fig. 3.13 RT-DES faster than real-time simulation experimentation

identified through the experiments. Harper [64] provides such a framework for real-time decision support in the context of the emergency department. Here the decision-makers are clinicians and managers.

The decision making can also be automatic (incorporated in an algorithm) and be used to make changes to a physical system. This is often referred to as symbiotic simulation. According to Aydt et al. [70], symbiotic simulation is the close coupling of a physical system and a computer model. Such simulations can benefit from the faster than real-time RT-DES approach. This is especially true for simulations of assembly lines and production facilities where decision making could be automated. For example, Onggo et al. [71] propose a hybrid modeling architecture for symbiotic simulation which includes data acquisition to receive data from the physical system, simulation and optimization models for faster than real-time experimentation, and an actuator that relays the results to the physical system.

3.8 Simulation of Comprehensive Systems

Baocun Hou, Bo Hu Li, Yang Liu

3.8.1 Connotation

Simulation of comprehensive systems is an important means of development and application of a comprehensive system, and it is becoming one of the research highlights in the field of modeling and simulation technology. Comprehensive systems mostly behave as continuous-discrete mixed, qualitative and quantitative mixed systems. They pose new challenges to modeling and simulation technology not only because they are large in size, complicated in composition and imperfect in knowledge but also because their behaviors are fuzzy, uncertain, difficult to qualify, and characterized with self-adaptivity, evolution, chaos, emergence, and gaming. It is our belief that the modeling and simulation technology of a comprehensive system is a kind of modeling and simulation technology which integrates the new-generation information communication technology, the new-generation artificial intelligence technology and the modern modeling and simulation technology with the specialized technology in the field of comprehensive system application, aiming at optimizing the overall performance of comprehensive system modeling, simulation operation, and result analysis/processing [72].

3.8.2 The Technical Framework

The technical framework of modeling and simulation technology for a comprehensive system is mainly composed of three kinds of technology, i.e., the modeling theory and methods for a comprehensive system, the simulation system technology

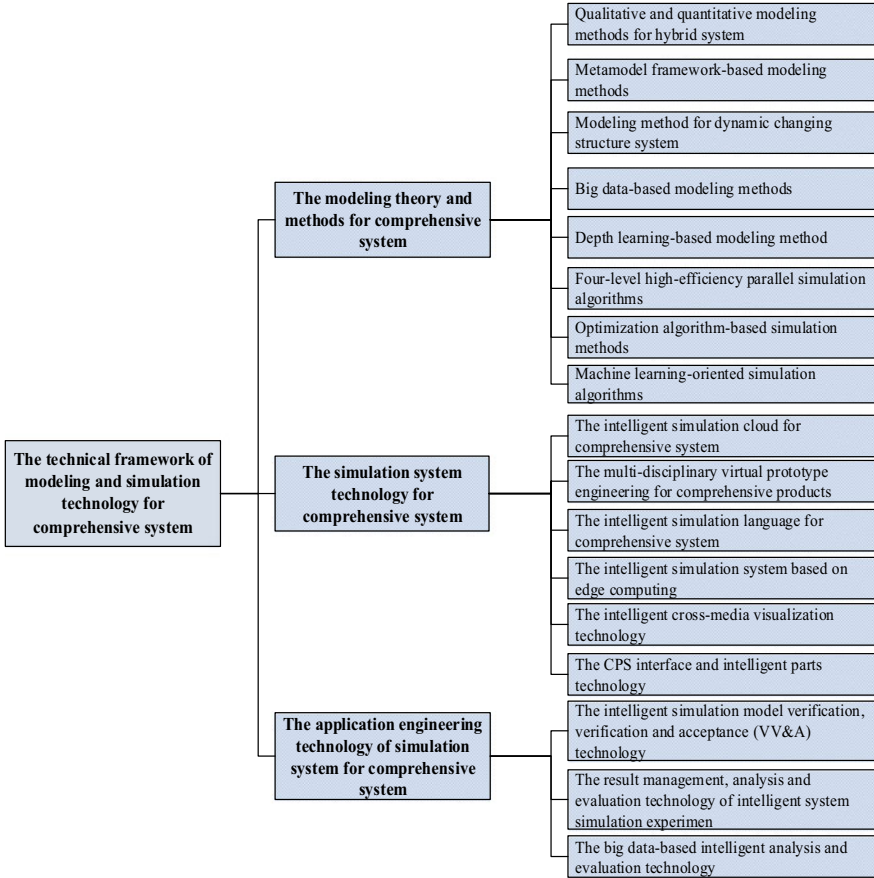


Fig. 3.14 Technical framework of modeling and simulation technology for a comprehensive system

for a comprehensive system, and the application engineering technology of simulation system for a comprehensive system, as shown in Fig. 3.14.

3.8.3 Key Technologies

The key technologies included in the technical framework of modeling and simulation technology of a comprehensive system are as follows.

The modeling theory and methods for comprehensive systems

The modeling theory and methods for comprehensive systems covers qualitative and quantitative modeling methods for a hybrid system, metamodel framework-based modeling methods, modeling method for dynamic changing structure system,

big data-based modeling methods, depth learning-based modeling methods, four-level high-efficiency parallel simulation algorithms, optimization algorithm-based simulation methods, and machine learning-oriented simulation algorithms, etc. [72].

- (1) The qualitative and quantitative modeling methods for a hybrid system include the qualitative and quantitative unified modeling methods, that is, those studies oriented to the modeling theory and methods for system top-level description and sub-domain description; the quantitative and qualitative interactive interface modeling, i.e., those studies oriented to transformation of the quantitative and qualitative interactive data into the structure and format required by the qualitative model and the quantitative model; the quantitative and qualitative time advance mechanism, namely those studies oriented to the time coordination and advance mechanism of the quantitative and qualitative models.
- (2) The metamodel framework-based modeling methods are the research on integrated simulation and modeling methods for multi-disciplinary, heterogeneous and emerging comprehensive systems through the top-level abstraction of the metamodel. It mainly includes multi-disciplinary unified modeling method based on meta modeling, i.e., the unified modeling method for continuous, discrete, qualitative, quantitative, and other multi-disciplinary models in a comprehensive system; the meta-model-based modeling method for comprehensive adaptive systems, namely the integrated simulation modeling method for perception, decision making and interaction between various system components in comprehensive adaptive systems.
- (3) The modeling method for dynamic changing structure systems mainly studies the dynamic changeability of the contents, ports, and connections of dynamic changing structure system model, in support of the overall modeling of the system structure in dynamic change.
- (4) The big data-based modeling methods: The high complexity of a comprehensive system's mechanism makes it difficult to build the system principle model on the mechanism (in an analytical manner), but renders it necessary to simulate its internal mechanism through a large number of experiments and application data. The big data-based modeling methods are a group of methods which enable an effective simulation of a comprehensive system with an unclear mechanism through massive observation and application data. The main research scope covers data-based reverse design, data-based neural network training and modeling, and data clustering-based modeling.
- (5) The depth learning-based modeling methods: In the environment of a comprehensive system, the data that can be collected and put to use grow explosively. At the same time, the neural network that can learn and evolve based on AI-based deep learning and human brain simulation can provide evolutionary support for the development and application of modeling and simulation oriented to a comprehensive system.
- (6) The four-level high-efficiency parallel simulation algorithms: In order to make full use of the super parallel computing environment to accelerate the problem

simulation of comprehensive systems, it is necessary to study high-efficiency four-level parallel simulation algorithms, including the job-level parallel method for large-scale simulation, the task-level parallel method for members of the simulation system, the model-level parallel method for members of the Federation, and the thread-level parallel method based on a comprehensive model solution.

- (7) The optimization algorithm-based simulation methods: Multi-sample iterative simulation is carried out based on optimization algorithms.
- (8) The machine learning-oriented simulation algorithms as an important driving force for the development of a comprehensive system to intelligence, machine learning, an important research field of artificial intelligence application, has evolved into a huge pedigree. How to effectively use machine learning to simulate and model comprehensive systems will become a new research orientation of great significance.

The simulation system technology for a comprehensive system

The simulation system technology for a comprehensive system includes the intelligent simulation cloud for comprehensive system, the multi-disciplinary virtual prototype engineering for comprehensive products, the intelligent simulation language for a comprehensive system, the intelligent simulation system based on edge computing, the intelligent cross-media visualization technology, and the CPS interface and intelligent parts technology.

- (1) The intelligent simulation cloud for a comprehensive system [73] is a new high-performance intelligent simulation mode that is based on ubiquitous network (including the Internet, Internet of Things, narrowband Internet of things, Internet of Vehicles, mobile Internet, satellite network, space-ground integrated information network, future Internet, etc.), being service-oriented and networked. Based on the idea of cloud computing, it evolves in response to application demands, integrating, and developing three kinds of technology, that is, (1) the existing networked modeling and simulation technology, (2) the emerging information technology such as cloud computing, Internet of Things, SOA, intelligent science, efficient computing and big data, and (3) the expertise for application fields. By virtualizing and servitization, it turns all kinds of simulation resources and capabilities into a cloud-based service pool, which are coordinated and optimized in terms of management and operation so that users can, via the network, terminals and cloud simulation platforms, gain access to such high-performance resources and capabilities for completion of various activities throughout the full life cycle of intelligence simulation [74].
- (2) The multi-disciplinary virtual prototype engineering for comprehensive products [75] is a kind of system engineering with a virtual prototype as the core and modeling and simulation as the means. Based on the integrated support environment, it is applied to optimize the five factors, that is, human, organization, business management, technology and data as well the four flows, i.e., information, knowledge, control, and service within the whole system of

comprehensive product development throughout the full life cycle. Its main research contents include the multi-stage unified modeling methods for virtual prototype engineering, the comprehensive decision making and simulation evaluation technology, the comprehensive management and prediction methods and the multi-disciplinary virtual prototype engineering platform.

- (3) The intelligent simulation language for a comprehensive system [76] is a simulation software system oriented to the modeling and simulation of a comprehensive system. Its main features are as follows: (1) the model descriptive form, which is composed of symbols, statements, and grammatical rules of the simulation language, and is very similar in terms of the model description to the original form of the system model under study; (2) the experiment description, which is composed of experimental operation statements similar to macroinstructions, and some ordered control statements; (3) it has rich parameterized and componentized simulation algorithm library, function library, and model library. It enables system researchers to focus on the comprehensive system simulation problem itself, therefore greatly reducing the software programming and debugging work related to modeling and simulation. Based on this simulation language, advanced simulation languages can be further developed for various special fields (such as military system-of-systems confrontation, multi-disciplinary virtual prototype simulation, etc.). Its main research contents involve the intelligent simulation language architecture, description specifications of simulation language for models and experiments, intelligent compilation and execution framework of simulation language based on simulation computer, etc.
- (4) The intelligent simulation system based on edge computing refers to the integrated intelligent simulation system that is aimed to optimize the overall performance of “system modeling, simulation operation and result analysis / processing.” Oriented to two kinds of simulation users (high-end modeling of comprehensive systems and on-demand provision of high-performance simulation cloud service), it enables three kinds of simulation (mathematics, human in the loop, hardware in the loop / embedded simulation) by integrating emerging computer technology (such as cloud computing, Internet of Things, big data, service computing, and edge computing), modern modeling and simulation technology, and supercomputer system technology. Its main research content involves computer system architecture, independent and controllable basic hardware / software, etc.
- (5) The intelligent cross-media visualization technology [77] mainly includes the GPU group-based parallel visualization system technology and the virtuality-reality fusion technology. The GPU group-based parallel visualization system technology involves the data organization and scheduling technology of large-scale virtual scene, the two-level parallel rendering technology based on multi-computer and multi-core technology, the high-efficiency visualization technology of amorphous objects in a comprehensive environment, and the real-time dynamic global illumination technology.

- (6) The CPS interface and intelligent parts technology: It mainly includes the research of the CPS interface technology and the R&D of simulation-specific acceleration components based on big data and artificial intelligence algorithms.

The application engineering technology of simulation system for a comprehensive system

The engineering technology of simulation system application for a comprehensive system mainly includes the intelligent simulation model verification, verification, and acceptance (VV&A) technology, the result management, analysis, and evaluation technology of intelligent system simulation experiment, and the big data-based intelligent analysis and evaluation technology.

- (1) The intelligent simulation model verification, verification, and acceptance (VV&A) technology [76] mainly includes such VV&A technologies as are applicable throughout the full life cycle and to the whole system, all hierarchical levels, all personnel, and all-round management.
- (2) The result management, analysis, and evaluation technology of intelligent system simulation experiment [76]: It mainly includes the simulation experiment data acquisition technology, the simulation experiment data analysis and processing technology, the simulation experiment data visualization technology, the intelligent simulation evaluation technology, and the benchmark technology (including two kinds of users and three kinds of simulation).
- (3) The big data-based intelligent analysis and evaluation technology [72] mainly includes the big data integration and cleaning technology, the big data storage and management technology, the big data analysis and mining technology, the big data visualization technology, the big data standards and quality system, and the big data security technology.

3.8.4 Development Trend

With the rapid development of Internet of Things, big data, cloud computing, high-performance computing and artificial intelligence, as well as their deep integration with modeling and simulation technology and specialized technologies for application fields, the modeling and simulation technology for a comprehensive system is developing to become more digitalized, high-efficient, networked / cloud-based, intelligent, service-oriented, and ubiquitous.

(1) Digitalized

Based on the Internet of Things and empowered with intelligent gateway, intelligent sensor and other means, the full life-cycle activities of the modeling and simulation for a comprehensive system are being digitized as a result of deep integration of the technologies used for collection, transmission, processing, and application of digital

information with simulation modeling, simulation systems, and the supporting technology as well as the engineering technology of simulation application.

(2) High-efficient

The deep integration of high-efficiency (high performance, high reliability, high energy saving, high availability) computing technology with simulation modeling, simulation systems, and the supporting technology as well as the engineering technology of simulation application is promoting the modeling and simulation of comprehensive systems to become highly efficient in the full life-cycle activities.

(3) Networked/Cloud-based

The deep integration of network communication, cloud computing, edge computing technology and simulation modeling, simulation systems and the supporting technology as well as the engineering technology of simulation application is making the modeling and simulation of a comprehensive system realize the networked / cloud-based full life-cycle activities.

(4) Intelligent

The deep integration of intelligent science and technology (brain science, cognitive science, artificial intelligence technology) with simulation modeling, simulation systems, and the supporting technology as well as the engineering technology of simulation application is adding intelligence to the modeling and simulation of comprehensive systems in the full life-cycle activities.

(5) Service-oriented

The deep integration of service computing (expression, discovery, construction, operation, evaluation, etc.) technology with simulation modeling, simulation systems and the supporting technology as well as the engineering technology of simulation application is promoting the modeling and simulation of comprehensive systems to turn service-oriented in the full life-cycle activities.

(6) Ubiquitous

The deep integration of ubiquitous computing (the integration of computing, communication and network technologies, and the integration of information space and physical space into a whole) technology with simulation modeling, simulation systems, and the supporting technology as well as the engineering technology of simulation application is making the modeling and simulation of comprehensive systems ubiquitous in the full life-cycle activities.

3.8.5 Application

The modeling and simulation technology for a comprehensive system is widely applied to the full life-cycle activities of a comprehensive system in the fields of the national economy, people's livelihood, national security, and so on. Our team has

developed some typical applications based on the research in respect of virtual prototype collaborative design simulation, intelligent simulation based on big data and cross-media reasoning, intelligent simulation based on virtual reality, and other related technologies. Examples are as follows:

(1) Application of the virtual prototype-based multi-disciplinary design and simulation technology in collaborative design of comprehensive products

In the field of intelligent manufacturing, the collaborative design of comprehensive products is realized through the virtual prototype-based multi-disciplinary design and simulation technology. A simulation application system for the virtual prototype collaborative design of intelligent aircraft is mainly composed of the control system model, the multi-body dynamics model, the hydraulic model, and other multi-disciplinary heterogeneous models. The aircraft shall be capable of monitoring its own flight parameters and status indicators, automatically predicting faults and adjusting flight status. Therefore, its capability of automatically anticipating faults and flight status is difficult to be described by a traditional quantitative model. It requires the use of qualitative rules. For this reason, we have built a simulation application system for the virtual prototype collaborative design of intelligent aircraft based on the modeling and simulation technology of a comprehensive system. The main work is as follows: In the stage of system construction, the internal static structure and the dynamic behavior logic of the comprehensive system are modeled at the top level of the system to eliminate the differences between various multi-disciplinary heterogeneous models in the system for the purpose of unified modeling. After the top-level modeling, a graphic model of the automatic prediction of faults and flight status is built on the basis of the fuzzy cause and effect diagram, with fuzzy concepts defined and modeled. In the meantime, the membership cloud model is trained by the training algorithm for the membership cloud model reverse generator. During the operation of the system, the simulation component model of a comprehensive system will be solved by the qualitative and quantitative joint approaches via the quantitative solution engine and qualitative reasoning engine that are driven by the service simulation engine.

(2) Application of multimodal data-based intelligent simulation analysis in Intelligent City

In the field of Intelligent City, the modeling and simulation technology is applied to realize the intelligent control of urban traffic light systems. The “green wave band” is the multi-point control technology of traffic lights, which can avoid congestion and save traffic time through intelligent control of signal lights. The “green wave band” uses the intelligent signal system, which senses the traffic flow through the underground coils, and automatically adjusts via computer the interval time of the traffic lights to reasonably distribute the signal period and prioritizes the traffic flow at the intersections. In the process of building a city’s green wave bands, the intelligent simulation and analysis are made based on multimodal data acquired through satellite navigation, monitoring cameras, traffic stations, sensor coils, etc., so that priority will be given to the installation and configuration of various

equipment in the green wave band system for the specific road layout to provide real-time analysis data for the urban signal system.

(3) Application of big data-based intelligent analysis in the medical field for disease early warning and prediction

In the field of intelligent health care, it has been made possible preliminarily to have an intelligent early warning and prediction of diseases based on the modeling and simulation technology for medical big data applications. At present, there are 70 million pieces of related infectious disease information on the national platform. Based on the ten million-level big data of major infectious diseases such as hepatitis B, tuberculosis and AIDS screening and queues, big data intelligence and cross-media reasoning technologies are used to establish a multi-factor analysis model and an intelligent visual analysis platform for the spread, evolution, and intervention of the “three diseases,” so as to enable intelligent early warning and prediction.

(4) Application of crops intelligent monitoring based on VR and M&S technology

In the field of intelligent agriculture, the independent growth of virtual crops is intelligently simulated by using the independent intelligent technology and the virtual reality technology, making it possible to reduce the research time and costs, and improve the quality and yield of crops through simulation and virtualization of the phenomena and process of crop growth. A geometric, physical, and behavior simulation is made of the object through the intelligent behavior modeling technology of virtual object, so as to improve the intelligent, social, diverse, and interactive fidelity of virtual object behavior. And the human–environment fusion technology in the virtual environment is used to realize the high-resolution 3D display, orientation tracking, gesture tracking, data glove, haptic feedback, and sound positioning.

References

1. Hart-Davis A (EiC) (2016) *Science: the definitive visual guide*. Dorling Kindersley Limited, London, England
2. Klein J, Giglioni G, Francis Bacon. In: Zalta EN (ed) *The stanford encyclopedia of philosophy* (Fall 2020 Edition). <https://plato.stanford.edu/archives/fall2020/entries/francis-bacon/>
3. Brown JR, Fehige Y (2019) Thought experiments. In: Zalta EN (ed) *The stanford encyclopedia of philosophy* (Winter 2019 Edition). <https://plato.stanford.edu/archives/win2019/entries/thought-experiment/>
4. Brown JR, Stuart MT (2020) Thought experiments. Oxford Bibliographies. <https://www.oxfordbibliographies.com/view/document/obo-9780195396577/obo-9780195396577-0143.xml>
5. King P (2020) *Learn to think using thought experiments: how to expand your mental horizons, understand metacognition, improve your curiosity, and think like a philosopher*. PKCS Media

6. Vedantham S (2020) Thought experiments: an inexpensive tour of the entire universe. Kindle Edition
7. Davis PK et al (2018) Priority challenges for social-behavioral research and its modeling, Santa Monica, Calif.: RAND
8. Sliva A, Neal Really S, Blumstein D, Peirce G (2019) Combining data-driven and theory-driven models for causality analysis in sociocultural systems. In: Davis PK, O'Mahony A, Pfautz J (eds) Social-behavioral modeling for complex systems. Wiley, Hoboken, pp 311–36
9. Halberstam D (2002) The best and the brightest. Modern Library
10. Wikipedia-thought experiment. https://en.wikipedia.org/wiki/Thought_experiment
11. Financial Crisis Inquiry Commission (2011) Financial crisis inquiry report. Washington, D. C.: GPO
12. Qin A (2020) China raises coronavirus death toll By 50% in Wuhan. New York Times, April 17. Accessed at <https://www.nytimes.com/2020/04/17/world/asia/china-wuhan-coronavirus-death-toll.html>
13. Cartwright N (2004) Causation: one word, many things. *Philos Sci* 71(5)
14. Halpern JY (2016) Actual causality. The MIT Press, Cambridge
15. Pearl J (2009) Causality: models, reasoning, and inference. Cambridge University Press, Cambridge
16. Pearl J, Mackenzie D (2018) The book of why: the new science of cause and effect. Basic Books, New York
17. Davis PK et al (2020) A complex systems agenda for influencing policy studies. WR-1326, Santa Monica, CA: RAND
18. Epstein JM (2019) Inverse generative social science—what machine learning can do for agent- based modeling. In: Davis PK, O'Mahony A, Pfautz J (eds) Social-behavioral modeling for complex systems. Wiley, Hoboken
19. Epstein JM, Axtell RL (1996) Growing artificial societies: social science from the bottom up. MIT Press, Cambridge
20. Lawrence JA, Edward Smith H (1991) The role of JSC engineering simulation in the apollo program. *Simulation* 57(1):9–16
21. Wooding CH et al (1973) Apollo experience report—simulation of manned space flight for crew training. NASA TN D-7112
22. Forrester JW (1961) Industry dynamics. Cambridge, Massachusetts
23. Forrester JW (1969) Urban dynamics. Wright Allen Press, Cambridge
24. Meadows DH, Jørgen R, Meadows DL (2004) The limits to growth : the 30-year update. Chelsea Green, White River Junction
25. Turner G (2008) A comparison of the limits to growth with thirty years of reality. CSIORO, Canberra
26. Buis A (2020) Study confirms climate models are getting warming projections right. *Global Climate Change*, Jan 9
27. Oppenheimer M et al (2019) Discerning experts; the practices of scientific assessment for environmental policy. University of Chicago Press, Chicago
28. Dörner D (1997) The logic of failure: recognizing and avoiding errors in complex situations. Perseus Books, Cambridge
29. Rouse WB (2020) Failure management: malfunctions of technologies, organizations and society. In: Rouse WB, Boff KR (2005) Organizational simulation. Wiley-Interscience
30. Rouse WB, Boff KR (2005) Organizational simulation. John Wiley & Sons
31. Rouse WB (2019) Human-centered design of model-based decision support for policy and investment decisions. In: Davis PK, O'Mahony A, Pfautz J (eds) Social-behavioral modeling for complex systems. Wiley, Hoboken, pp 798–808
32. Ferguson NM et al (2020) Impact of non-pharmaceutical interventions (NPIs) to reduce COVID-19 mortality and healthcare demand. Imperial College, London

33. Huang NE, Fangli Q, Ka-Kit T (2020) A data-driven model for predicting the course of COVID-19 epidemic with applications for China, Korea, Italy, Germany, Spain. medRxiv.
34. Vardavas R et al (2020) The health and economic impacts of nonpharmaceutical interventions to address COVID-19: a decision support tool for state and local policymakers, TLA173-1
35. Davis PK, Popper SW (2019) Confronting model uncertainty in policy analysis for complex systems: what policymakers should demand. *J Policy Complex Syst* 5(2):181–201
36. Enserink M, Kai K (2020) Mathematics of life and death: how disease models shape national shutdowns and other pandemic policies. *Science*, Mar 25
37. Marchau VAWJ, Walker WE, Bloemen PJT, Popper SW (eds) (2019) *Decision making under deep uncertainty: from theory to practice*. Springer, Cham
38. Lempert RJ, Popper SW, Bankes SC (2003) *Shaping the next one hundred years: new methods for quantitative long-term policy analysis*. RAND Corporation, Santa Monica
39. Ören TI, Zeigler BP (1979) Concepts for advanced simulation methodologies. *Simulation* 32(3):69–82
40. Ören TI (2009). Modeling and Simulation: a comprehensive and integrative view. In: Yilmaz L, Ören TI (eds) *Agent-directed simulation and systems engineering*. Wiley series in systems engineering and management, Wiley-Berlin, Germany, pp 3–36
41. Ören TI (2009b) Uses of simulation. Chapter 7. In: Sokolowski JA, Banks CM (eds) *Principles of modeling and simulation: a multidisciplinary approach*, (All chapters by invited contributors). Wiley, New Jersey, pp 153–179
42. Barton RR (2013) Designing simulation experiments. In: Pasupathy R, Kim S-H, Tolk A, Hill R, Kuhl ME (eds) *Proceedings of the 2013 winter simulation conference*. <https://informatics-sim.org/wsc13papers/includes/files/029.pdf>
43. Kleijnen JPC (2008) *Design and analysis of simulation experiments*. Springer, New York
44. Law AM (2007) *Simulation modeling and analysis*, 4th edn. McGraw-Hill, New York
45. Hunter JS, Naylor TH (1970) Experimental designs for computer simulation experiments. *Manag Sci* 16(7) (March 1970)
46. Zeigler BP, Muzy A, Kofman E (2019) *Theory of modeling and simulation—discrete event & iterative system computational foundations*, 3rd edn. Academic Press
47. Giambiasi N, Escude B, Ghosh S (2000) GDEVS: a generalized discrete event specification for accurate modeling of dynamic systems, simulation: *Trans Soc Model Simul Int* 17(3):120–134
48. Simon HA (1972) *The architecture of complexity*. Facets of systems science. Springer
49. Traoré MK (2020) Multi-perspective modeling and holistic simulation: a system-thinking approach to very complex systems analysis. In: Mittal S, Tolk A (eds) *Complexity challenges in cyber physical systems: using modeling and simulation (M&S) to support intelligence, adaptation and autonomy*. System Engineering and Management. ISBN: 9781119552468, pp 83–110
50. Burns AJ, Kopp RE (1961) Combined analog-digital simulation. In: AFIPS 61, proceedings of the eastern joint computer conference, pp 114–123
51. Cellier FE (1979) *Combined continuous/discrete system simulation by use of digital computers*. Doctoral dissertation, ETH Zurich
52. Shanthikumar JG, Sargent RG (1983) A unifying view of hybrid simulation/analytic models and modelling. *Oper Res* 31(6):1030–1052
53. Mustafee N, Katsaliaki K (2020) Classification of the existing knowledge base of OR/MS research and practice (1990–2019) using a proposed classification scheme. *Comput Oper Res*. <https://doi.org/10.1016/j.cor.2020.104920>
54. Harper A, Mustafee N, Feeney M (2017) A hybrid approach using forecasting and discrete-event simulation for endoscopy services. In: *Proceedings of the 2017 winter simulation conference*, December 3–6, Las Vegas, NV. IEEE
55. Harper A, Mustafee N (2019) Proactive service recovery in emergency departments: a hybrid modelling approach using forecasting and real-time simulation. In: *Proceedings of the 2019 ACM SIGSIM conference on principles of advanced discrete simulation*. ACM

56. Fishwick P, Mustafee N (2019) Broadening participation in modelling. In: Proceedings of the 2019 winter simulation conference, December 8–11, National Harbor, Maryland. IEEE
57. Mustafee N, Powell JH (2018) Towards a unifying conceptual representation of hybrid simulation and hybrid systems modelling. In: Proceedings of the operational research society simulation workshop 2018 (SW18), March 19–21, Worcestershire, UK. UKORS
58. Mustafee N, Brailsford S, Djanatliev A, Eldabi T, Kunc M, Tolk A (2017) Purpose and benefits of hybrid simulation: contributing to the convergence of its definition. In: Proceedings of the 2017 winter simulation conference, December 3–6, Las Vegas, NV. IEEE
59. Mingers J, Brocklesby J (1997) Multimethodology: towards a framework for mixing methodologies. *Omega* 25(5):489–509. [https://doi.org/10.1016/S0305-0483\(97\)00018-2](https://doi.org/10.1016/S0305-0483(97)00018-2)
60. Mustafee N, Powell JH (2018) From hybrid simulation to hybrid systems modelling. In: Proceedings of the 2018 winter simulation conference (WSC), December 9–12, Gothenburg, Sweden. IEEE, pp 1430–1439
61. Mustafee N, Bischoff EE (2013) Analysing trade-offs in container loading: combining load plan construction heuristics with agent-based simulation. *Int Trans Oper Res* 20(4):471–491. <https://doi.org/10.1111/itor.12017>
62. Brailsford S, Eldabi T, Kunc M, Mustafee N, Osorio AF (2019) Hybrid simulation modelling in operational research: a state-of-the-art review. *Eur J Oper Res* 278(3):721–737. <https://doi.org/10.1016/j.ejor.2017.10.025>
63. Powell JH, Mustafee N (2017) Widening requirements capture with soft methods: an investigation of hybrid M&S studies in healthcare. *J Oper Res Soc* 68(10):1211–1222. Palgrave, Macmillan. <https://doi.org/10.1057/s41274-016-0147-6>
64. Mustafee N, Powell JH, Harper A (2018) RH-RT: a data analytics framework for reducing wait time at emergency departments and centres for urgent care. In: Proceedings of the 2018 winter simulation conference. IEEE, pp 100–110
65. Harper A (2021) A hybrid modelling framework for real-time decision-support for urgent and emergency healthcare. University of Exeter Business School. Ph.D. thesis
66. Harper A, Mustafee N (2019) A hybrid modelling approach using forecasting and real-time simulation to prevent emergency department overcrowding. In: Proceedings of the 2019 winter simulation conference (WSC). IEEE, pp 1208–1219
67. Fujimoto RM (2000) Parallel and distributed simulation systems. Wiley, New York
68. Bryant RE (1977) Simulation of packet communication architecture computer systems (No. MIT/LCS/TR-188). Massachusetts Institute of Technology, Cambridge
69. Chandy KM, Misra J (1981) Asynchronous distributed simulation via a sequence of parallel computations. *Commun ACM* 24(4):198–206
70. Jefferson DR (1985) Virtual time. *ACM Trans Progr Lang Syst* 7(3):404–425
71. Aydt H, Turner SJ, Cai W, Low MYH (2009) Research issues in symbiotic simulation. In: Proceedings of the 2009 winter simulation conference. IEEE, pp 1213–1222
72. Onggo BS, Mustafee N, Smart A, Juan AA, Molloy O (2017) Symbiotic simulation system: hybrid systems model meets big data analytics. In: Proceedings of the 2018 winter simulation conference. IEEE, pp 1358–1369
73. Li BH, Chai XD, Zhang L, Li T, Qing DZ, Lin TY et al (2018) Preliminary study of modeling and simulation technology oriented to neo-type artificial intelligent systems. *J Syst Simul*
74. Ju RS, Yang M, Zhong RH, Liu XC, Zhou Y, Huang KD (2013) Summary of service oriented modeling and simulation. *Syst Eng Electron* 35(7):1539–1546
75. Li T, Li BH, Chai XD (2012) Layered simulation service description framework oriented to cloud simulation. *Comput Integr Manuf Syst* 18(9):2091–2098
76. Ling XQ, Huang XD, Li BH (2013) Collaborative modeling and simulation platform with its key techniques. *J Syst Simul* 25(06):1264–1269
77. Li BH, Chai XD, Li T, Hou BC, Lin TY, Xing C et al (2012) Research on high-efficiency simulation technology of comprehensive system. *J China Acad Electron Inf Technol*

-
78. Li BH, Chai XD, Hou BC, Li T et al (2009) Networked modeling & simulation platform based on concept of cloud computing—cloud simulation platform. *J Syst Simul.* 21 (17):5292–5299
 79. Financial Crisis Inquiry Commission (1963) *Industrial dynamics*. MIT Press, Cambridge
 80. Merriam-Webster-experiment: <https://www.merriam-webster.com/dictionary/experiment>
 81. Bishop M (1998) An epistemological role for thought experiments. In: Shanks N (ed) *Idealization IX: idealization in contemporary physics*. Poznan studies in philosophy of science and humanities Bookseries, Rodopi
 82. Davis PK (2022) Supporting social science and management areas, in this volume
 83. (forthcoming) SCS modeling and simulation book of knowledge
 84. Collins-hindcast: <https://www.collinsdictionary.com/dictionary/english/hindcast>



Simulation as Experience to Enhance Three Types of Skills

4

Tuncer Ören , Umut Durak , Ernest H. Page, Andreas Tolk , and Saikou Y. Diallo 

Abstract

Enhancing skills is one of the main reasons for the use of simulation. This chapter of the SCS M&S Body of Knowledge looks mainly at training. The use of simulators, often referred to as virtual simulation, is described and followed by the use of constructive simulation systems, where all relevant entities of interests are simulated. Examples from the defense sector are given, but also health care and emergency management. Finally, a section describes the various options of live simulation, where trainees with their operational equipment are stimulated by simulated inputs.

Keywords

Modeling and simulation · Training simulation · Live simulation · Virtual simulation · Constructive simulation

T. Ören (✉)

University of Ottawa, Ottawa, ON, Canada

e-mail: toren@uottawa.ca

U. Durak

German Aerospace Center, Cologne, Germany

e-mail: Umut.Durak@dlr.de

E. H. Page

The MITRE Corporation, McLean, VA, USA

e-mail: epage@mitre.org

A. Tolk

The MITRE Corporation, Charlottesville, VA, USA

e-mail: atolk@mitre.org

S. Y. Diallo

Old Dominion University, Norfolk, USA

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2023

T. Ören et al. (eds.), *Body of Knowledge for Modeling and Simulation*,

Simulation Foundations, Methods and Applications,

https://doi.org/10.1007/978-3-031-11085-6_4

4.1 Types of Simulation Techniques for Experience

Tuncer Ören.

Gaining experience is another pillar of modeling and simulation—along with experimentation. Simulation-based experience is used to gain/enhance three types of skills or for entertainment purposes.

4.1.1 Simulation Experience for Training

As outlined in Table 4.1, simulation experience can be used for training get/enhance three types of skills.

1. *Virtual simulation* (i.e., use of simulators or virtual simulators) is used to enhance motor skills to gain proficiency of use of equipment such as an airplane, a tank, or a car. In virtual simulation, real people use virtual equipment in virtual environments, hence the term ‘virtual simulation’.
2. *Constructive simulation* (or gaming simulation such as war gaming, peace gaming, international relations gaming, and business gaming) is used to enhance decision-making and/or communication skills. In constructive simulation, simulated people use simulated equipment in a virtual environment and real people get experience by interacting with the simulation system.
3. *Live simulation* is used to gain/enhance operational skills by getting real-lifelike experience in a controlled environment. Live simulation is used in such diverse areas as military exercises as well as for the training of health specialists. In live simulation, real people use imitation (or virtual or dummy) equipment in the real world” [1].

Table 4.1 Use of simulation for training to get/enhance three categories of experience-based training

Purpose of training	Category of simulation	Type of simulation
To enhance motor skills	Virtual simulation	– Simulators – Virtual simulators
To enhance decision-making and/or communication skills	Constructive simulation	– Gaming simulation – Wargaming – Peace gaming
To operate by getting real-lifelike experience opportunities in a controlled environment	Live simulation	– Single platform simulation – Integrated multiplatform simulation – Live simulation of systems of systems; federations of federations; hyper federations

4.1.2 Simulation Experience for Entertainment

Use of simulation in entertainment covers a large application area. The techniques used in this area have an important overlap with serious games [2] and education [3].

4.2 Virtual Simulation

Umut Durak.

DoD Modeling and Simulation Glossary [4] defines the term “virtual” as an entity or data that stems from a modeled or simulated representation of the actual system. At the same reference “virtual simulation” is defined as a simulation involving real people operating simulated systems. While there are various application areas, training simulators are typical examples of this type.

Training is one of the major application areas of simulation. While the virtual simulation is not exclusive for training, the area is widely driven by training simulators. Application domains render a wide range from flight training to surgery training. Flight simulators can be taken as a representative example for virtual simulation. Allerton [5] and Page [6] present a comprehensive historical perspective for flight simulators.

The flight simulators date back to the beginning of the twentieth century. During the early 1900s, the idea was to design a truly ground-based trainer that can provide the students with an understanding of how to fly an aircraft. Sanders Teacher [7] depicted in Fig. 4.1 was one of the earliest flight simulators, a virtual simulation, that was constructed from actual aircraft mounted on a universal joint facing toward an existing wind. It was simulating the real aircraft using the response of its aerodynamic responses to the prevailing wind. Due to the irregular nature of the wind, Sanders Teacher did not bring the expected success. Almost at the same time, Antoinette company designed its training rig, called “Tonneau Antoinette”—Antoinette’s barrel where the instructor is manually changing the attitude of the aircraft to train students for using controls. Edward Link is the founder of modern flight simulators. In the late 1920s, he designed his Link trainer, commonly known as the *Blue Box* applying compressed air to tilt the cockpit and drive the gauges for aircraft instruments [5]. Over half a million pilots were trained in Blue Boxes only during the Second World War. Since then, flight simulators are indispensable parts of flight training (Fig. 4.2).

Flight simulators have also been utilized by the research community since the Apollo mission [8]. They are invaluable ground-based test facilities in developing and experimenting with advanced concepts and conducting human factor research. Some of the well-known early examples are ATTAS Ground-Based Simulator from German Aerospace Center (DLR), [8, 9] NASA Crew Vehicle Systems Research Facility in Ames Research Center [10], and Visual Motion Simulation and Cockpit Motion Facility from Langley Research Center [11]. Relatively recent research

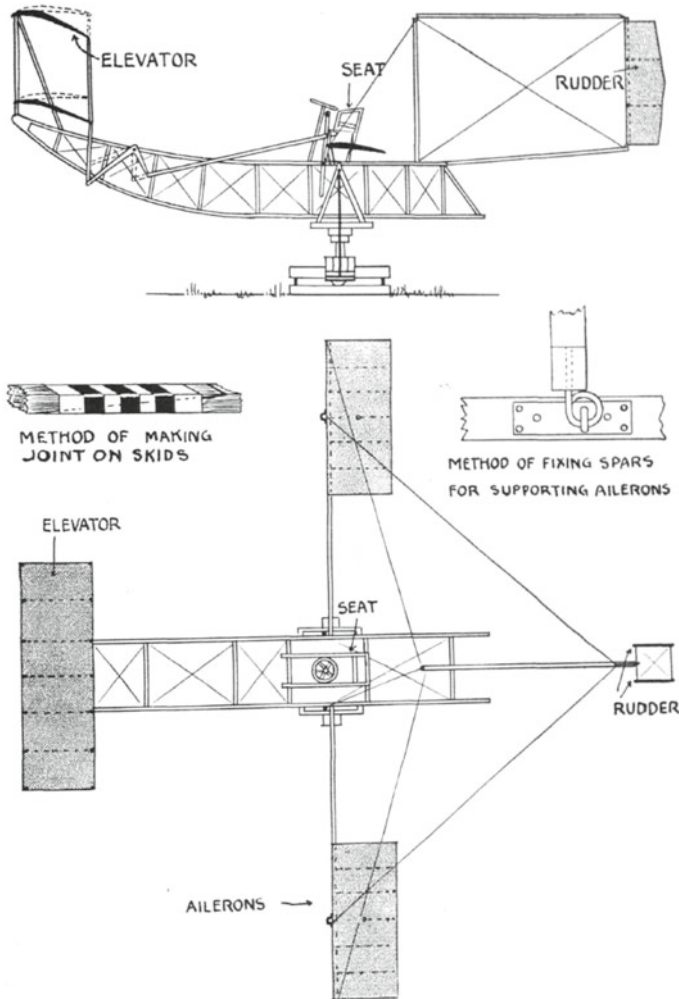


Fig. 4.1 Sanders teacher (Reprinted from [7])

flight simulators that are currently in operation include Air Vehicle Simulator (AVES) of German Aerospace Center (DLR) [12], HELIFLIGHT from the University of Liverpool [13], NASA Ames Vertical Motion Simulator [14], and SIMONA of Delft University of Technology [15].

Flight simulator experiments have long been used to evaluate the handling qualities of air vehicles [16–20]. Besides quantitative analysis, qualitative ratings can also be collected from the pilots by incorporating flight simulators in the air vehicle design process. Today, it is also a common practice to test avionic systems on the ground with flight simulation in respective integration test facilities before testing them in real flight. Examples include Boeing 777 System Integration Lab



Fig. 4.2 Link trainer (the blue box)

[21] and F-35 flight simulators [20] in which support engineering laboratories from multi-ship simulations to HIL testing.

The human-in-the-loop nature of virtual simulation brings several challenges some of which are real-time constraints and high fidelity visual, aural, and motion queuing requirements [5]. Furthermore, the wide utilization of simulators for training the operators commonly in safety-critical domains, such as aviation, brought certification and accreditation regulations of virtual simulators. Examples could be the ICAO manuals of criteria for the qualification of flight simulation training devices [22, 23].

Typical organization of virtual simulators consists of the dynamic model of the device, its operational environment, a visual system, a sound system, possibly a motion system, a realistic human device interface, and an instructor operator station. Figure 4.3 presents the architecture of the rotorcraft simulation at the German Aerospace Center (DLR) Air Vehicle Simulator (AVES). It depicts the simulator components that correspond the above-explained virtual simulator organization. Examples could be the Air Vehicle Model as the dynamic model of the device and its operational environment, or the Cockpit and the Control Loading System for human device interface.

Virtual simulation—as a very powerful tool—is applicable to many disciplines. For example, several examples from the medical community are given in the virtual simulation section of a book on healthcare simulation [24]. Another book focuses on virtual simulation in nursing education [25]. Rozenblit et al., contributed to very specific applications of virtual simulation in health care, such as “haptic guidance system for computer-assisted surgical training [26].

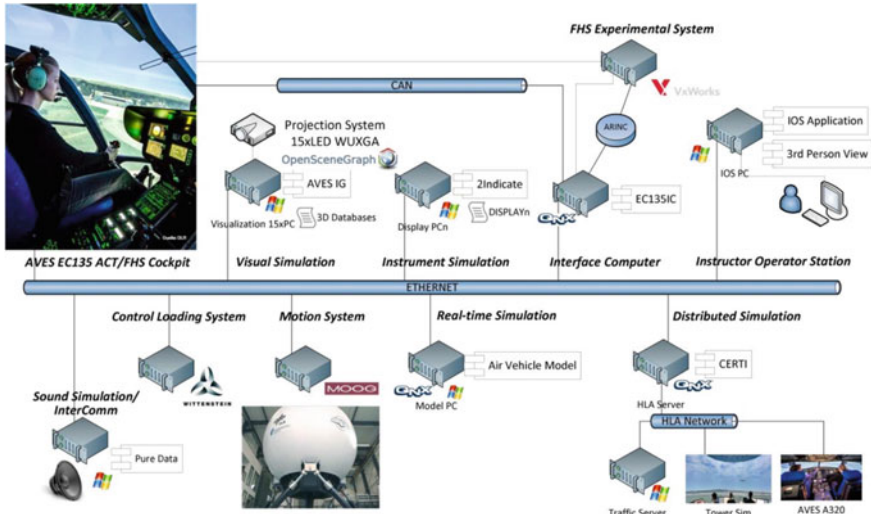


Fig. 4.3 German aerospace center (DLR) air vehicle simulator (AVES) architecture

A completely different successful application is virtual simulation in manufacturing process of high-speed trains [27]. Another study focuses on “issues related to human effectiveness within embedded virtual simulations (EVS) for training.” [28].

4.3 Constructive Simulation

Ernest H. Page and Andreas Tolk.

According to the introduction of this section, constructive simulation is a computer simulation in which people, equipment, and environment are simulated, although real people get experience and enhance their decision-making and/or communication skills by interacting with the simulation and potentially with each other. Why do we call this category constructive?

According to Dictionary.com, the adjective *constructive* has four different meanings, which are (1) helping to improve; promoting further development or advancement (opposed to destructive); (2) of, relating to, or of the nature of construction; structural; (3) deduced by inference or interpretation; inferential; and (4) Law: denoting an act or condition not directly expressed but inferred from other acts or conditions. How do these definitions help to understand what constructive simulation is?

We surely hope that all categories of simulation will be helpful (as in (1)), and we also assume that we use good practices when constructing our simulations (as in (2)), and we furthermore assume that we can exclude the law context (as in (4)). This leaves us with the third definition to be the most meaningful in our context:

Constructive simulation is simulation that has been deduced by inference or interpretation, leading to a computer simulation of the system of interest. However, as every simulation is based on a model which is a simplification and abstraction of the real system within the constraints of the experimental frame this is also not a clear distinction.

The best way to think about a constructive simulation is that all relevant processes of the systems of interest, including possible users of those systems and their decision processes, are simulated. There will be trainees outside of the simulation, but everything important reacting to their inputs is simulated. There are obvious close relations to virtual simulation, particularly when the communication with the simulation uses everyday operational equipment of the trainees to do so, but while virtual simulators are focused more on individuals or small teams, constructive simulation is used to train decision makers, or leadership team group members. With the provided examples, this should become clearer.

4.3.1 Examples for Constructive Simulation

Constructive simulations provide reactive representations of complex, often socio-technical systems. Examples are battles with hundreds or thousands of weapon systems engaging along a very long front to train military headquarters, or highly complex flight tracking systems connecting continents to train crews of aviation administration. We can also think about a city to support city planners in many details, from coordination of traffic lights to maximize traffic throughput to better placing of restaurants or grocery shops.

The application domains are not limited to training, but this simulation category is also used for analysis and planning, as the simulations often ran faster than real time, so that many repetitions are possible, allowing not only for sensitivity analysis, but also for exploratory modeling [29].

In some cases, game technology allowed for very realistic interfaces, as games are designed to immerse players into the simulated processes, at least as an observer. When, e.g., an avatar must provide a situation report via a video-feed, game technology gives simulated person a face. Another example is the use of video streams to display the discoveries of a drone when flying over an area. In some cases, the whole simulation can be a serious game [30].

Computer-Assisted Exercises

Within the defense domain [31], the use of constructive simulation systems to train command posts and headquarters on various military levels and different scales is well-established. Training the headquarter personnel of a military organization is challenging. Headquarters oversee coordinating and synchronizing the activities by their subordinated commands by means of Command and Control. The activities are often captured in form of the so-called O-O-D-A loop, which stands for observe, orient, decide, and act. The loop starts with observation, which triggers all the following activities. Observation, in this context, is not just what can be seen visually,

but includes data and information available through other sensing mechanisms, communications reports, and so forth. Within orientation, the headquarter works toward understanding the observation, i.e., understanding where the opposing forces are, where their own forces are, in which state they are in, and other relevant conditions. Based on the results of the orientation, the staff comes to a decision by evaluating alternative courses of action and choosing the best option. The decision is followed by actions putting this decision to work, usually by generating orders for the subordinated commands and forces. This leads to the next loop, as the result of such action is observed, leading to new orientation, decision, and action, and so forth.

Providing realistic input to the training staff as well as adequate feedback on developments and effects is the main challenge to be addressed for an exercise. The most realistic way is to use all the subordinated forces to provide the input and react to the commands (see the next subsection on live simulation and maneuvers), but even when not taking an enemy into consideration, the pure manpower needed quickly becomes overwhelming regarding costs and administration: A military battalion has 400 to 1000 soldiers, and lots of equipment, if you go up to the division level, this number grows to 6000 up to 25,000 soldiers. Only in war situation, such numbers of personnel will be available.

Constructive simulation systems that realistically simulating all the soldiers, weapon systems, equipment, communication, movement, attrition, etc., were therefore always required by the armed forces. While in early computer-assisted exercise only the training personal had access to the simulation, and they did feed the resulting reports into the Command and Control systems used by the trainees (as well as taking the orders and feeding them into the simulation), the technical integration of simulation systems to directly communicate with the Command and Control systems allow for significant reduction of personnel needed for an exercise [32]. By the early 1990s, Command Post Exercises (CPXs) shifted from predominantly live events to a mixture of live, virtual, and constructive, with the predominant amount of the force structure represented in constructive simulation, as discussed in Sect. 4.4.2).

Wargaming

At its core, wargaming is a tool for exploring human decision making, particularly in environments with incomplete and imperfect information. Perla [33] defines wargaming as "...a warfare model or simulation whose operation does not involve the activities of actual military forces, and whose sequence of events affects and is, in turn, affected by the decisions made by players representing the opposing sides."

Typically, wargames are strategically focused. During a wargame, players may discover the need to make unanticipated decisions in order for the game to progress. According to [34], there can also be an educational component to a wargame. Experience has shown that players learn from each other while participating in the wargame. Most players find the exchanges of ideas and information that occur during a wargame to be professionally rewarding.

Some common types of wargames are [35]:

- *Table-Top Exercise.* A table-top exercise is a discussion-based wargame where players sit at tables and interact with one another to address the key issues of the wargame. While not specifically structured as a turn-based game, facilitators will often cause players to consider issues in a particular order, to determine the relationship between specific decisions or actions.
- *Workshop.* Workshops involve subject-matter experts (SMEs) gathered to discuss a problem. Workshops have a narrow, discrete focus, and often serve as an input to follow-on events.
- *Inductive game.* Inductive games begin without a pregame concept. With inductive games, the concept is discerned after analyzing game data for patterns. This type of gaming is used early in the concept development process and makes use of open-ended brainstorming styles during the event.
- *Deductive game.* In contrast, deductive games begin with general game ideas to be tested, followed by observations collected during the game to support or refute the initial game hypothesis. This type of gaming is used later in the concept development process, after the concept is more fully developed. This is used during course of action (COA) analysis or to test a plan prior to execution.
- *Scenario-based game.* This technique presents players with a specific scenario, which is used to guide the course of the wargame while the players examine a particular strategic problem or issue. Scenario-based games, starting with present-day conditions, can be used to “take an intellectual walk into the future.” Based on a sponsor’s requirements, the wargame may be based on a specified scenario.
- *Alternative futures game.* An alternative futures wargame involves presenting the participants with two or more scenarios of a plausible future. Players are asked to determine key indicators that would signal that the future represented by the scenario might be emerging. In contrast to the scenario-based game, an alternative future game starts in the future and works backward to the present. Game results often include identifying both unique and common indicators from across several scenarios. Toward the end of game play, the players may be asked to identify what they believe is the most plausible future based on game play.
- *Single-sided game.* A single- or one-sided game includes one player cell, with the opposition furnished by a control group that presents scripted scenario injects.
- *1½-sided game.* A 1½-sided game also includes one player cell, with the opposition furnished by a control group, but with scenario injects developed during game execution, versus pre-scripted, to force the players to wrestle with specific decisions related to game objectives.
- *Two-sided game.* Two-sided games involve two, separate, competing player cells. The two sides play by rules that vary from restrictive to entirely free play. Player decisions from each cell are adjudicated, with results presented to the players and used to inform subsequent game play.

- *Multisided game.* Games may be designed with several competing cells. These games are referred to as multisided, or by the actual number of sides (e.g., “three-sided”). The rules of conduct for multisided games can be significantly more complex than in a two-sided game due to the number of possible interactions between the various player cells.

Wargaming has a rich history in the US military. The Naval aviation community’s deliberate and aggressive use of wargames during the 1930s is often credited with the defeat of the Japanese carrier force at the Battle of Midway in the Second World War [36].

As captured by [37], the reason for the need to rely increasingly on computational support by simulations lies in the complexity of evolving concepts. As observed in the epilogue, “such challenges of complexity and emergence have been addressed in language and constructs of their times by authors from Sun Tzu to Clausewitz. However, due to jointness, networking, long-range fires, precision weapons, special operations, gray-area operations, and other developments, the complexity of military planning and operations has increased substantially. Traditionally separated domains merge into each other on the modern battleground, including the new elements of cyber. This requires true creativity of wargamers, but also the support by simulation in two main categories: (1) providing a decision space with the necessary complexity to be representative of the real-world scenario, and (2) allowing for exploratory analysis of situations.” This observation is not only true for military application, but also in many other domains, from business to health care.

Healthcare

Medical training applications are typically focusing on individual or small group skills required, e.g., in the surgery room, so that more examples can be found in life and virtual training domains [24]. Examples given in [38] demonstrate that constructive simulations of hospitals or even hospital groups are increasingly used by decision makers in wargaming-like scenarios. Most of these exercises are currently conducted in support of finding better strategies, e.g., how to better organize a hospital, how many Intensive Care Units to run versus using the space alternatively for more hospital beds, but also how to better align with other hospitals in the region, e.g., by providing rarely needed specialty treatment not in all facilities, etc. However, conducting such efforts also leads to better problem understand and communication among these decision makers, as described among others in [39].

In addition, biology and health care are using constructive simulation increasingly to understand the complex interplay of interest, as described among others in [40] and [41]. Such models were applied in the recent fight against the COVID-19 pandemic as well.

Emergency and Crisis Management

Šimic [42] provides examples on how the crisis staff community can learn from the military community when using constructive simulation systems as a collaborative

learning tool. Again, the constructive simulations of crisis situations comprising various personnel and equipment applicable to cope with these crises are in the center of the approach. Such applications are not yet as common as in the military domain, but the numbers of examples of successful use of constructive simulation are growing, as the scholastic community starts to engage more in this new application field [43]. Here are some examples.

Planning and evacuation [44] is an endeavor that requires the orchestration of many organizations that usually do not work or train together daily. First responders, like fire departments and police, should know how to direct people from the danger zones. Neighbored cities must align their evacuation plans to avoid that one community uses a route for evacuation that another community reserves for quick responder movement and therefore blocks for standard traffic. Providing a realistic environment is key for success. Hurricane evacuations in coastal areas must react timely to flooding within the evacuation area: Are the evacuation routes still passable? Can hospitals still be reached? A good constructive simulation provides such challenges to the trainees in form of geospatial representations as well as simulating the related effects when simulating actions.

Emergency with mass casualties [45] often require the collaboration of different organizations, from security forces safeguarding the emergency area, hospitals being prepared to accept certain numbers of patients and communicating their preparedness, ambulance crews with the best equipment are directed to the victims in need and to the hospitals prepared to accept patients, etc. Using constructive simulation to prepare these crews, e.g., in case of a high-speed train accident or a bomb attack on a crowded sports event, will increase the awareness, preparedness, and communication between the various organizations.

4.3.2 Serious Games

In his seminal paper, [46] defines a serious game as “a mental contest, played with a computer in accordance with specific rules that use entertainment to further government or corporate training, education, health, public policy, and strategic communication objectives.” While traditional computer games are focusing on the story, presenting art, and software development challenges, serious games add the challenge of pedagogy. In other words, the main purpose is no longer entertainment, but teaching a skill needed for real-life tasks.

As in many of the other sections, the defense domain was among the strong supporters [47], and serious gaming has a secure place since 2006 as the Serious Games Showcase & Challenge (SGS&C) during the annual Interservice/Industry Training, Simulation and Education Conference (IITSEC), but as shown in [48] many other industry and business domains are applying serious games for team building, communication, interpersonal skills, negotiation skills, creativity, as well as learning innovation, risk management, health and safety, and more.

One of the reasons to use gaming technology for training and education was the development of immersive, realistic graphics. While traditional simulations used maps and dashboard-like displays, serious games used 3D displays of the story. In the military domains, Virtual Battle Space from the company Bohemia Interactive Simulations gained a lot of support for its realistic display and integration of artificial intelligence to create challenging opponents [49]. As the entertainment industry provided significant resource for the development of these technologies for entertainment, the serious gaming community could focus on the pedagogic elements.

Serious gaming is today contributing to the body of knowledge with dedicated journals, such as *Simulation and Gaming* by SAGE Publications, but also in special issues of other simulation publications. As gaming and simulation are also taught together in high schools, this interdependence will likely become even more important in the future.

4.3.3 Additional Application Domains of Interest

Within the previous section, we already extended the application domains beyond the traditional domain of defense and emergence response. Those were geared toward command centers that are in charge to orchestrate many different processes, be it the coordination of a military operation or the synchronization of humanitarian relief operations after a hurricane.

With the increasing awareness how much complexity effects are decision-making abilities, the use of constructive simulations to educate decision makers in coping with complexity becomes more and more popular. Rouse [39] proposes that decision makers must be able to immerse into the complex problem space. They also need to have controls at hand easy enough to use them intuitively, but also powerful enough to evaluate their various options. Rouse uses the picture of a “decision makers’ flight simulator” to help to look at alternative courses of action and their results. This kind of policy evaluation has been identified as a helpful use of constructive simulation in various domains and received great popular visibility during the COVID-19 pandemic, where various models were used to evaluate the effectiveness of interventions, see, e.g., [50] or [51].

Another topic of interest is climate change and its effects. Constructive models can link the effects of climate change and the effects of possible policy decisions and show the result to provide immediate feedback to the decision makers. Using the latest insights from computational social sciences, artificial societies can now be created that let agents act based on insights how humans perceive situations and make decisions from cognitive and psychological perspectives. This opens even more application domains, such as questions of social justice, equity, and others.

4.4 Live Simulation

Saikou Y. Diallo, Andreas Tolk.

Modeling is as old as humanity. It is one of the ways we experience the world, teach our children, entertain ourselves, and maintain a link with our past. From the humanities to social sciences and the arts, models embody theories, ideas, concepts, and act as an effective means of communication. Live models should be quite familiar as we use them frequently to navigate our day. From parents modeling behaviors for their children, to actors role-playing in movies and the Theater, modeling is omnipresent in our life.

Simulation modeling is essentially a mapping activity. Each map varies by person, activity, subject-matter, and objective. Mapping can be objective or subjective. It can be biased and instantaneous or follow strict rules over a long period of time. In some instances, mapping can be automated, discovered, or projected and in others it relies on ingrained and evolved processes that we are not even fully aware of. Mapping helps us identify, classify orient, solve problems, and make decisions.

In the context of this section, we understand live simulation as acting upon these concepts, purposefully placing individuals of groups into a realistic scenario to expose them to experiences needed to better cope with upcoming challenges while minimizing the danger that comes with practicing the behavior in its real setting, such as battle, emergencies, and accidents.

4.4.1 Examples for Live Simulation

Live simulation or live modeling is mostly used to train, practice, and rehearse with the goal of gaining and enhancing skills through lifelike experiences in a controlled environment. The argument can be made that this is likely the oldest form of applied simulation borne out necessity and a pragmatic approach required to survive a sometimes harsh environment. For example, we have archeological evidence that hunters practiced their aiming skills to ensure successful hunting expeditions from the earliest days on. According to Flavius Josephus, as quoted in [52], the Roman Army successfully used organized training regiments at the beginning of our common era: *“Their battle-drills are no different from the real thing... It would not be far from the truth to call their drills bloodless battles, their battles bloody drills.”* This gave the Romans a clear advantage, as many of the peoples they faced during their conquests learned the art of war on the job: they gathered community members fit for military service for battle in case of need and often had little chance against the well-practiced drills.

It is no surprise that live simulation in a controlled environment became practice in many domains. Surgeons practiced on cadavers to learn where important organs are in the human body, so they did not create additional harm by wounding important organs during the surgery on life patients. Architects experimented with different building styles before building houses. Today, fire drills help building

occupants learn how to best escape a real fire; active shooter exercises help schools and other organizations to be better prepared for such emergency. As a rule, live simulation is best suited when people need to gain skills in a domain but cannot train on the job because it is impractical, too dangerous, or too expensive. In the next section, we present a few examples.

Military and Law Enforcement Domain

The military examples of drills and maneuvers are provided among others [31]. As defined above, in these drills and maneuvers, soldiers behave like they should do in real warfare, but they are not in danger by real opposing forces. There may be an opposing force in place, but their goal is to educate the soldiers, not to kill them in battle. They are today common practice in all defense forces and, as the next section shows, are increasingly made more realistic by borrowing methods from other simulation categories.

Law enforcement personnel of local, state, and federal organizations practice their skills in live simulation environments, such as provided in the Federal Law Enforcement Training Centers (FLETC) in Glynco, GA. FLETC has multiple firearms ranges, including an indoor range complex with many separate firing points. Law enforcement officers do not only train the use of their firearms, but they are also put into realistic lifelike scenarios like hostage situations where they must decide of to engage using their weapons. Furthermore, driver training ranges, a physical techniques facility, explosives range, fully functional port of entry, and numerous other structures support the training effort.

Medical and Healthcare Domain

Medical simulation is also making use of life simulation. Cardiopulmonary resuscitation (CPR) is a life-saving skill as many people as possible should have, and training for it requires drills in a controlled environment. Usually, mannequins are used to practice CPR to be prepared in case of need. A recently conducted study [53] indicates that simulation-based CPR training programs are effective in improving knowledge and performing CPR, as well as in decreasing stress of CPR in clinical nurses. Medical personnel are also trained by human actors who describe their simulated symptoms to the trainees, described beside other means by [54]. The live actors are using increasingly supplemental devices to provide additional simulated hints on the source of the symptoms. An additional example is the use of simulated hospitals, such as used in the Center for Simulation and Education Enhancement of the Health branch of the University of California in Davis. In a realistic setting, nurses practice CPR, assist in operation rooms, and have to take care of multiple patients with different symptoms and needs at the same time.

Management, Theater, and Entertainment

Many organizations provide employees regular training on how to behave legally and ethically to provide a safe and comfortable workplace for everyone. The training usually involves some live simulation where the employees are exposed to representative scenarios and vignettes that model key situations. In addition,

employees might participate in might participate in team-building exercises that at their workplace or during a retreat. These exercises often involve some form of live simulation where role-playing in a simulated context helps make a point or stress key aspects of teamwork.

Similarly, theater is used in schools to teach children social, cultural, and historical aspects of life in society. In those cases, children are often asked to play archetypical roles where they embody an ethos. The group lead by a teacher is then asked to critique, explain, and discuss the situations represented in the play thereby simulating key situation and hopefully understanding the key concepts embodied in the play. The entertainment industry also relies heavily on live simulation to practice key aspects of their craft such as positioning, lighting and live rehearsal to ensure an effective delivery of the final product.

4.4.2 Live Simulation in the Context of LVC Architectures

The term “live simulation” became popular as part of the so-called live–virtual–constructive federation. It can be traced back to the 1993 Report of the Defense Science Board [55], which was tasked in a summer study to evaluate the impact of advanced distributed simulation on readiness, training, and prototyping for the armed forces of the United States. The defense community provides the following definitions, such as in the United States Department of Defense in the Modeling and Simulation Glossary (2014). The definitions are captured in Table 4.2 as well. They are applicable to the broad simulation community and by no means limited to the defense domain.

Live—A simulation involving real people operating real systems. Military training events using real equipment are live simulations. They are considered simulations because they are not conducted against a live enemy.

Virtual—A simulation involving real people operating simulated systems. Virtual simulations inject a Human-in-the-Loop into a central role by exercising motor control skills (e.g., flying jet or tank simulator), decision-making skills (e.g., committing fire control resources to action), or communication skills (e.g., as members of a C4I team).

Constructive—A simulation involving simulated people operating simulated systems. Real people stimulate (make inputs to) such simulations but are not involved in determining the outcomes. A constructive simulation is a computer program. For example, a military user may input data instructing a unit to move and to engage an enemy target. The constructive simulation determines the speed of

Table 4.2 Live–virtual–constructive simulation categories

	Live	Virtual	Constructive
People	Real	Real	Simulated
Equipment	Real	Simulated	Simulated
Environment	Controlled	Virtual	Synthetic

movement, the effect of the engagement with the enemy, and any battle damage that may occur. These terms should not be confused with specific constructive models such as computer-generated forces (CGF), a generic term used to refer to computer representations of forces in simulations that attempts to model human behavior. CGF is just one example model being used in a constructive environment. There are many types of constructive models that involve simulated people operating simulated systems.

The environment is real but controlled for live simulation. For simulators, i.e., virtual simulators, the environment is virtual: implemented as a situated environment for the simulator and presenting it often in detail as an immersive representation. This immersive component is not always needed for constructive simulations but can be utilized, e.g., when the video-stream of a simulated drone through synthetic terrain and simulated enemy systems must be shown to live audiences.

When building a composition of simulation systems from all three categories, some additional challenges must be addressed.

- LVC architectures require real-time simulations. Live participants with their real systems in their controlled but real environment cannot execute tasks faster than real-time, nor is it tolerated having to wait for slower-than-real-time simulation results too often, as that will contradict the training objectives.
- The controlled environment of life simulation cannot be controlled by virtual or constructive simulation results. A bridge in the real world is not destroyed by a simulated attack, and virtual bombing campaigns do not leave craters in landing strips of airports. However, the use of extended reality (see next paragraph) can support better alignment.

4.4.3 Enhancing Live Simulation with Augmented Reality

The recent developments in extended reality (XR), such as defined by Hillmann [56], which combines immersive techniques of virtual reality (VR)—lifelike display of virtual environments—and augmented reality (AR)—the lifelike display of synthetic objects overlaid into real world—are blurring the limits between the three simulation categories. Life soldiers participating in a maneuver wearing AR glasses displaying enemy troops that are simulated in a constructive simulation is an example using elements from all three categories. These recent developments support the original goal of live simulation: to gain or enhance operational skills by real-lifelike experiences in a controlled environment. We are just using new simulation-based methods to enhance this even further in all domains, as surgeons operating on realistically reacting cadavers or mannequins benefit from these developments as much as air traffic controllers having to react to hazardous situations realistically presented to them via all senses.

4.4.4 Ethical Constraints and Conclusion

We close this section with a discussion on the ethical considerations associated with live simulation. It is important to recognize that humans operate on a physical, cognitive, and socioeconomic spectrum. Modelers share the responsibility of ensuring that models capture the spectrum of stakeholders impacted by the solution they are designing. This is especially true for live simulations since it relies on the physical and cognitive abilities of the participants. As a result, it is essential to follow an ethical, inclusive, and traceable modeling process where bias is recognized, managed, and disclosed. The modeling process for live simulation requires obtaining information from stakeholders to gain insight into their perspectives on a situation. It is also essential to consider those who are impacted by the models as part of the stakeholder group. Without those considerations, the simulation will miss key insights and will not be usable by some segments of the stakeholder community. An inclusive design tied to ethics in modeling is even more important to ensure that people of all spectrums (visual, hearing, etc.) are accounted for in the definition of the problem space. We also reiterate the need to be multidisciplinary and seek expertise where it exists. Modeling is a team effort that requires subject-matter experts, modelers, and analysts to collaborate. The teams need to be diverse and inclusive as much as possible to account for the diversity in experience and background found in society. We recommend actively seeking diversity as another way to account for the bias inherent in modeling.

Although live simulation predominantly has been seen as training of physical skills, the domain is widening to social and even cognitive skills. New technologies can provide a nearly lifelike experience, immersive environments so real that our brains can hardly cope with the difference of live simulation and reality. This leads to tremendous opportunities, but also a great responsibility of the simulation experts supporting such endeavors to avoid manipulation of the participating individuals or groups.


References

1. Ören TI (2009) Modeling and simulation: a comprehensive and integrative view. In: Yilmaz L, Ören TI (eds) *Agent-directed simulation and systems engineering*. Wiley Series in Systems Engineering and Management, Wiley-Berlin, Germany, pp 3–36
2. NRC (National Research Council) (1997) *Modeling and simulation: linking entertainment and defense*. The National Academies Press, Washington, DC. <https://doi.org/10.17226/5830>. <https://www.nap.edu/download/5830>
3. NRC (National Research Council) (2011) *Learning science through computer games and simulations*. The National Academies Press, Washington, DC. <https://doi.org/10.17226/13078>. <https://www.nap.edu/download/13078>
4. DoD Modeling and Simulation (M&S) Glossary (1998), DoD 5000.59-M, DoD
5. Allerton D (2009) *Principles of flight simulation*. Wiley
6. Page RL (2000) Brief history of flight simulation. *SimTecT 2000 Proceedings*, pp 11–17
7. Haward DM (1910) The sanders “Teacher”. *Flight*, pp 1006–1007

8. Duda KR, de Saint Phalle R, Schroeder C, Johnson M, Fill T, Miller B, Ventura A, Siegmann A, Mudgal K, Morrill R, Gravina S (2020) Development of a lunar lander simulator: commemorating apollo and looking to the future. In AIAA Scitech 2020 Forum, p 0371
9. Klaes S (2000) ATTAS ground based system simulator -an update. AIAA modeling and simulation technologies conference and exhibit, Denver, CO
10. Sullivan B, Soukup P (1996) The NASA 747–400 flight simulator: a national resource for aviation safety research. AIAA flight simulation technologies conference, San Diego, CA
11. Smith R (2000) A description of the cockpit motion facility and the research flight deck simulator. AIAA modeling and simulation technologies conference and exhibit, Denver, CO
12. Duda H, Gerlach T, Advani S, Potter M (2013) Design of the DLR AVES research flight simulator. AIAA modeling and simulation technologies (MST) conference, Boston, MA
13. White MD, Padfield GD (2006) The use of flight simulation for research and teaching in academia. AIAA atmospheric flight mechanics conference and exhibit, Keystone, CO
14. Advani S, Giovannetti D, Blum M (2002) Design of a hexapod motion cueing system for NASA ames vertical motion simulator. AIAA modeling and simulation technologies conference and exhibit, Monterey, CA
15. Stroosma O, van Paassen R, Mulder M (2003) Using the simona research simulator for human-machine interaction research. AIAA modeling and simulation technologies conference and exhibit, Austin, TX
16. Muckler F, Obermayer R (1963) Performance measurement in flight simulation studies. AIAA heterogeneous combustion conference, Palm Beach, FL
17. van Gool M, Weingarten N (1981) Comparison of low-speed handling qualities in ground-based and in-flight simulator tests. AIAA 1st flight test conference. Flight test conference, Las Vegas, NV
18. Kiefer D, Calvert J (1992) Developmental evaluation of a centrifuge flight simulator as an enhanced maneuverability flying qualities tool. AIAA flight simulation technologies conference, New Orleans, LA
19. Anderson F, Biezd D (1998) A low-cost flight simulation for rapid handling qualities evaluations during design. AIAA modeling and simulation technologies conference and exhibit, Boston, MA
20. Landry L (2008) Application of modeling, simulation and labs to the F-35 program. AIAA modeling and simulation technologies conference and exhibit, Honolulu, HI
21. Lansdaal M, Lewis L (2000) Boeing's 777 systems integration lab. *IEEE Instrum Meas Mag* 3 (3):13–17
22. ICAO (2015) Manual of criteria for the qualification of flight simulation training devices, Volume I—Aeroplanes (9625–1). International Civil Aviation Organization, Montreal
23. ICAO (2012) Manual of criteria for the qualification of flight simulation training devices, Volume II—Helicopters (9625–2). International Civil Aviation Organization, Montreal
24. Levine AI, DeMaria Jr S, Schwartz AD, Sim AJ (eds) (2013) *The comprehensive textbook of healthcare simulation*. Springer Science and Business Media
25. Gordon RN, McGonigle D (eds) (2018) *Virtual simulation in nursing education*. Springer Publishing Company, New York, NY
26. Hong M, Rozenblit JW (2016) A haptic guidance system for computer-assisted surgical training using virtual fixtures. 2016 IEEE international conference on systems, man, and cybernetics (SMC), pp 002230–002235. <https://doi.org/10.1109/SMC.2016.7844570>
27. Wang W, Wu B, Hu YM, Li MY, Liu QY, He C (2012) Virtual simulation applications in manufacturing process of high-speed trains. *Adv Mater Res*, 622–623, 575–580. <https://doi.org/10.4028/www.scientific.net/amr.622-623.575>
28. Sottolare R, Roessing JJ (2014) The application of intelligent agents in embedded virtual simulations (EVS). In NATO technical report: improving human effectiveness through embedded virtual simulation: findings of task group HFM-165. NATO Science and Technology Organization. ISBN: 978–92–837–0181–1
29. Kwakkel JH, Pruyt E (2013) Exploratory modeling and analysis, an approach for model-based foresight under deep uncertainty. *Technol Forecast Soc Chang* 80(3):419–431

30. Curry J, Price T, Sabin P (2016) Commercial-off-the-shelf-technology in UK military training. *Simul Gaming* 47(1):7–30
31. Tolk A (2012) *Engineering principles of combat modeling and distributed simulation*. John Wiley & Sons Inc., Hoboken, NJ
32. Cayirci E, Marincic D (2009) *Computer assisted exercises and training: a reference guide*. John Wiley & Sons, Hoboken, NJ
33. Perla PP (1990) *The art of wargaming: a guide for professionals and hobbyists*. Naval Institute Press
34. United States Army War College (2015) *Strategic wargaming series handbook*
35. United States Naval War College (2013) *War Gamers' Handbook*
36. Mizokami K (2013) A brief history of naval wargames. Retrieved August 24, 2021, from <https://news.usni.org/2013/09/24/brief-history-naval-wargames>
37. Turnitsa CD, Blais C, Tolk A (2022) *Simulation and wargaming*. Wiley, Hoboken, NJ
38. Padilla JJ, Diallo SY, Armstrong RK (2018) Toward live virtual constructive simulations in healthcare learning. *Simul Healthc* 13(3):35–40
39. Rouse WB (2020) Understanding the complexity of health. *Syst Res Behav Sci* 1–7
40. Scharzt R (2008) *Biological modeling and simulation: a survey of practical models, algorithms, and numerical methods*. MIT Press, Cambridge, MA
41. DiStefano J III (2013) *Dynamic systems biology modeling and simulation*. Elsevier Inc, London, UK
42. Šimic G (2012) Constructive simulation as a collaborative learning tool in education and training of crisis staff. *Interdiscip J Inf Knowl Manag* 7:221–236
43. Kincaid JP, Donovan J, Pettitt B (2003) Simulation techniques for training emergency response. *Int J Emergency Manage* 1(3):238–246
44. Gan H-S, Richter K-F, Shi M, Winter S (2016) Integration of simulation and optimization for evacuation planning. *Simul Model Pract Theory* 67:59–73
45. Zouari A, Ghanem C, Dridi S, Abdelmoumen S, Daghfous M (2007) Simulation training. *Prehosp Disaster Med* 22(S1):s14–s15
46. Zyda M (2005) From visual simulation to virtual reality to games. *Computer* 38(9):25–32
47. Smith RD (2010) The long history of gaming in military training. *Simul Gaming* 41(1):6–19
48. Riedel JC, Hauge JB (2011) State of the art of serious games for business and industry0. 17th international conference on concurrent enterprising. IEEE Press, pp 1–8
49. van Diggelen J, Heuvelink A, Muller T, van den Bosch K (2010) Using artificial team members for military team training in virtual environments. NATO symposium on human modelling for military application, pp. P2–1–P2–5
50. Ferguson N, Laydon D, Nedjati Gilani G, Imai N, Ainslie K, Baguelin M, Dighe A, et al (2020) Impact of non-pharmaceutical interventions (NPIs) to reduce COVID19 mortality and healthcare demand. Imperial College, London, UK
51. Kerr C, Stuart R, Mistry D, Abeyesuriya R, Rosenfeld K, Hart G, George L (2021) Covasim: an agent-based model of COVID-19 dynamics and interventions. *PLOS Comput Biol* 17(7): e1009149
52. Chaliand G (1994) *The art of war in world history: from antiquity to the nuclear age*. University of California Press, Berkeley, CA
53. Sok SR, Kim JA, Lee Y, Cho Y (2020) Effects of a simulation-based CPR training program on knowledge, performance, and stress in clinical nurses. *J Contin Educ Nurs* 51(5):225–232
54. Zapko KA, Ferranto MLG, Blasiman R, Shelestak D (2018) Evaluating best educational practices, student satisfaction, and self-confidence in simulation: a descriptive study. *Nurse Educ Today* 60:28–34
55. Defense Science Board (1993) *Impact of advanced distributed simulation on readiness, training and prototyping*. Report ADA266125. Defense Science Board, Washington, DC
56. Hillmann C (2021) The history and future of XR. In: *UX for XR*. Design thinking. Apress, Berkeley, CA



Rodrigo Pereira dos Santos  and Esteban Walter Gonzalez Clua 

Abstract

This chapter of the SCS M&S Body of Knowledge describes scope, terminology, and applications of simulation in the context of gaming for health, education, business, transportation, environmental challenges, and sports.

Keywords

Modeling and simulation · Gaming · Serious games

5.1 Scope

Rodrigo Pereira Dos Santos, Esteban Clua.

According to Salen and Zimmerman [1], a game is a system in which players engage in fictional conflicts, operated by well-defined rules, resulting into an objective or quantifiable outcome. A digital game has the same definition, but it operates into a virtual world and environment. Typically, this system simulates different scenarios, inspired by real or completely fictional contexts. In any case, it is possible to assume that any game is and requires a simulation [2]. However, while entertainment games have as its main objective to entertain, a simulation system intends to create and operate a specific situation, not necessarily aiming to bring fun.

R. P. d. Santos (✉)

Federal University of the State of Rio de Janeiro, Rio de Janeiro 22290-255, Brazil
e-mail: rps@uniriotec.br

E. W. G. Clua

Universidade Federal Fluminense, Niterói 24210-346, Brazil
e-mail: esteban@ic.uff.br

Therefore, it is comprehensible that the terms “game”, “simulation”, and “simulation game” have been explored by researchers and practitioners over the years, even though a common sense on their precise meanings is a challenge for quite some time [3]. Some features are explored or even required in games and, consequently, in simulation game: (1) technological and mathematical elements, (2) arts and design concerns; (3) cultural aspects; (4) focus on application domains; and (5) game elements such as participants, roles, rules, competition, and cooperation [4]. Other features refine the definition of simulation games based on the simulation background: (1) real-world system representation, (2) rules and strategies for evolving flexible and variable activities; and (3) low cost of error with no critical consequences or losses [5].

The entertainment and engagement aspect of games brings important features related to the learning and understanding process. A game is interactive and requires actions of the player, who is the protagonist of the system. As such, the simulation only flows when a user is doing the required tasks and the player is also protagonist of his/her comprehension process, drastically enhancing the capacity of teaching [6]. For this reason, making simulation systems with game elements creates enhanced and powerful educational systems.

In some contexts, simulation and serious games can be seen as a combination of game features and simulation features [7]. Within this scope, gamification became an important topic, in which elements of games (e.g., rules, storytelling, challenges, etc.) are incorporated into “serious” or real case simulations or processes [8].

Despite any blurred frontier, the preparation of (and participation in) a simulation game can support a better understanding of a real-life situation, then enhancing participants’ knowledge and skills, even in the case of simulation for entertainment [9]. It is worth highlighting that the use of many game elements in simulation games varies according to the application domain. An example is presented in a recent systematic literature review on students’ experiences in learning clinical reasoning based on simulation games, in which self-developed instruments used to assess experiences were found, but most of them were poorly validated [10]. Moreover, results of another systematic review on the transportation domain show that the existing studies used to reproduce real environments through simulation games. To do so, such games combine achievement (challenges and points), immersion (role playing), and social (cooperation) resources [11]. As such, simulation games have been played by experienced gamers because such games require some time for familiarity to be not only accessible, but also enjoyable [12].

More recently, another important topic related to the field is receiving special attention: the creation of metaverses or Digital Twins. Due to the big number of tools and resources available for modeling virtual scenarios for games and simulations, the process of creating virtual clones of real elements (e.g., real cities, oil and gas platforms, vehicles, human bodies, etc.) is growing fast. This will allow the availability of complex simulations to many areas and application domains. NVIDIA Omniverse [13] and Microsoft Azure Digital Twins [14] are examples of powerful tools that are already commercially available. In the near future, it is expected that almost all factories, big corporations, and complex scenarios will have their virtual clone, allowing to simulate any kind of situation and process.

Given the evolving nature and challenges on development and use of simulation for the past decades, different areas have explored games in the topic, for example:

- “Organizational Simulation”, covering aspects from modeling and simulation to games and entertainment [15];
- “Ecosystem Simulation”, covering strategies to improve quality of life in the world based on analysis of trade-offs, e.g., a better environment and food production, despite the potential limitation of simplifying scenarios [16];
- “Discrete-Event Simulation”, covering research and learning on modeling and simulation (M&S) in queueing systems, not only with real application in biology, chemistry, physics, and statistics, but also considering entertainment [17];
- “Software Engineering Simulation”, covering education and training of software company management on teams, projects, products, and customers with the benefits of fun and entertainment, from conception to completion of the software development process [18, 19];
- “Information Systems Simulation”, covering simulation games for explaining and exploring case studies in some dynamic, contemporary actions such as digital marketing, given the previous successful adoption [20].

The potential of simulation games in those areas (and others) depends on active collaboration and interaction among players, especially due to the fact that humans do not learn from their own exposure to (or experience with) complex relationships from real-world scenarios [5]. In this direction, simulation games used to be more effective than other instructional methods, considering the intertwined affective and cognitive processes, so that its use in education is highly recommended to develop skills [21]. Moreover, augmented reality can enrich simulation games in their ability to connect educational content and the physical world, balancing and managing resources [22]. This combination is indeed confirmed by several studies published in the journal *Simulation and Games* from the end of 1960s [23]. A similar scenario was reported in the context of business simulation games from 1950s, in which seven key dimensions should be considered: realism, accessibility, compatibility, flexibility and scale, simplicity of use, decision support systems, and communication [24].

Finally, some effects of simulation game preferences relate to different game player profiles, e.g., age, gender, background, etc., [25]. Trade-offs between faithfulness to reality, simplicity, and fun factors should also affect players, for both card-based and automated approaches [18]. Other issues to be considered from game players are as follows: first adoption experience, objectives for use, achievements, information search, game evolution, etc., [26]. Additionally, simulation games are more effective if players should develop decision-making abilities for managing complex and dynamic situations [27]. In the engineering domain, for instance, their use maximizes the transferability of academic knowledge to the industry, since “what-if” analyses can explore different (sometimes non-feasible) solutions based on a “practice in safety” approach [28].

5.2 Terminology

Rodrigo Pereira Dos Santos, Esteban Clua.

Attributes [3]: It consists of characteristics or specific properties used to describe an element. For example, in the educational domain, some game attributes are player, conflict, rules, goal, artificial character, and educational character. In turn, some educational simulations attributes are a model of reality (system), dynamic-simplified-valid, and an educational purpose. Finally, as a combination, educational simulation game attributes are a simulation, players in competition or cooperation, rules, and educational character.

Concept [24]: A simulation game focuses on a specific area of an application domain. As such, a concept of simulation refers to such area. For example, in a business simulation game, a concept can focus on traffic management, advertising, sales, human resources, etc. This allows more precise analyses on players' behaviors based on mental models and cognitive mapping as well as simulation game effectiveness [29].

Design [11]: It is a step that focuses on the identification of the most relevant decisions to be implemented. In addition, simulation game design aims to improve solutions and cope with complexity. As such, designers should analyze the pre-defined goals as well as how to achieve them. For example, a multiplayer and cooperative simulation game explores shared situational awareness, whereas a single player simulation game explores achievement-based resources in order to stimulate individual efficiency.

Debriefing [9]: It refers to an important phase in the use of simulation games, in which participants are invited to link experiences from playing the simulation game and experiences from real-life situations. Some issues raised when designing a debriefing session are as follows: transfer of knowledge and skills, the simulation game's design, context of the participants, individual and collective learning, facilitator's knowledge of the simulation game, participants' perceptions of the relation between simulation game and real life, and type of process (stepwise or cyclical). Three phases are involved: description, analysis, and application [30].

Digital Twin [31]: It addresses a virtual and digital model from a real object, process, or complex scenario, capable of simulating and recreating different aspects of its functionalities. It involves a collection of relevant digital artifacts including domain data, behavioral descriptions, life cycle details, and interface information.

Gamification [8]: It involves incorporation of game elements, such as objectives, points, badges, challenges, storytelling, etc., into a process or system in order to include more engagement and lucidity for the participants.

5.3 Applications

Rodrigo Pereira Dos Santos, Esteban Clua.

Health: Simulation games influence behavioral determinants (e.g., knowledge, attitudes, skills, etc.). As such, improving quizzes with role-playing and simulation games stimulates behavioral determinants and intense interaction. Therefore, some important factors in the health domain are different formats, user customization, levels of difficulty, and feedback. Some applications: (1) In sexual health, real-world activities are mimicked through accurate depictions of steps in that process, and opportunities are provided for practice in a safe gaming environment [32], (2) in mental health, interventions included simulation games to improve communication with patients with disorders, as well as significant improvements in knowledge and skill scores post-intervention [33], and (3) in child health care, group interventions for the prevention of injuries in young children based on simulation games [34].

Education: Given the large number of studies on simulation game in the educational domain, four elements are pointed out as success key factors: the role of an instructor, the integration in the context of a course, the technical specifications covered in the course, and the practical experience based on intense collaborative work [35]. In this context, improvements on virtual learning environments to support remote education have been performed, as several institutions evolved their strategies, especially considering the COVID-19 pandemic [36]. This scenario opens opportunities for integrating simulation games in the development and improvement of knowledge, skills, and competencies [37]. In addition, students learn more in their experience in simulation games if there is a personalized communication agent [38]. This is somehow crucial to avoid some level of boredom and anxiety, which can negatively affect problem solving strategies [39].

Business: Referred as business simulation games (BSG), the goal is to explore real aspects of a business environment in order to assist beginners and experts in managing organizational activities and decision-making based on risk-free scenarios and on a view of strategic functions [40]. In this case, the use of an attractive and interactive approach to do so is relevant, including research on user experience and its effects on the participants from the perspectives of psychophysics, cognitive neuroscience, and computer science [37], even considering some natural simplification [41]. In addition, over a decade, supporting a virtual team member to recognize colleagues' knowledge, trust in colleagues' expertise, and coordinate knowledge remains as a challenge [42]. Some measures in this context comprise team performance, disposition to trust, and trust itself [43], sometimes depending on the team members' distribution [44].

Transportation: According to a recent literature review on the topic [11], the existing studies focus on implementing simulation games to cope with operational level issues regarding road and maritime transportation modes when improving freight transportation. From 40 studies analyzed in this context, simulation games

covered all decision-making levels, considering psychological (experience, perceived reality/learning/usefulness, fun, engagement, motivation, satisfaction, and attitude) and behavioral (efficiency and resilience) issues. Some scenarios investigated in the existing studies are supply chain and logistics operation management, development of a new area in a maritime port, shipping schedule and optimization, as well as profit maximization in road transportation.

Environment: Simulation game allow the investigation of coupled natural and human systems if exploring interaction between humans and the environment in which they live (or have interested in). For example, considering urban hydrology and air pollution, a player (e.g., mayor) can analyze people's actions and the built and nature environment, as well as their how such relations dynamically affect each other, in the geography and novel engineering curricula [45].

Sports: The assessment of intensity and energy cost of different modalities in sports is also explored in Simulation game, including different levels of difficulties as well as training and performance indications. For example, quantity/quality of exercise for developing and maintaining cardiorespiratory fitness ensuring a player's safety is addressed in a dance simulation game, driven by recommendations of reference authorities on the subject [46].





References

1. Salen K, Zimmerman E (2003) Rules of play—game design fundamentals. MIT Press
2. Joselli M, Zamith M, Valente L, Feijó B, Leta FR, Clua E (2014) A distributed architecture for simulation environments based on game engine systems. In: Visual computing (F. Rodrigues Leta). Augmented Vis Reality 4:41–61, Springer. https://doi.org/10.1007/978-3-642-55131-4_2
3. Sauv e L, Renaud L, Kaufman D (2010) Games, simulations, and simulation games for learning: definitions and distinctions. In: Educational gameplay and simulation environments: case studies and lessons learned (D. Kaufman and L. Sauv e), IGI Global, pp 1–26. <https://doi.org/10.4018/978-1-61520-731-2.ch001>
4. Xavier BL, Santos RP, Viana D (2020) Software ecosystems and digital games: understanding the financial sustainability aspect. In: Proceedings of the 22nd international conference on enterprise information systems, vol 2, ICEIS, pp 450–457, SciTePress. <https://doi.org/10.5220/0009794204500457>
5. Garris R, Ahlers R, Driskell JE (2002) Games, motivation, and learning: a research and practice model. Simul Gaming 33(4):441–467. <https://doi.org/10.1177/1046878102238607>
6. McGonigal J (2011) Reality is broken: why games make us better and how they can change the World. Penguin Books
7. Ruohom aki V (1995) Viewpoints on learning and education with simulation games. In: Simulation games and learning in production management (J.O. Riis). IFIP advances in information and communication technology, pp 13–25, Springer. https://doi.org/10.1007/978-1-5041-2870-4_2
8. Detering S, Dixon D, Khaled R, Nacke L (2011) From game design elements to gamefulness: defining “gamefication”. In: Proceedings of the 15th international academic mindtrek conference: envisioning future media environments, pp 9–15. <https://doi.org/10.1145/2181037.2181040>

9. Peters VAM, Vissers GAN (2004) A simple classification model for debriefing simulation games. *Simul Gaming* 35(1):70–84. <https://doi.org/10.1177/1046878103253719>
10. Havola S, Koivisto J, Mäkinen H, Haavisto E (2020) Game elements and instruments for assessing nursing students' experiences in learning clinical reasoning by using simulation games: an integrative review. *Clin Simul Nurs* 46:1–14. <https://doi.org/10.1016/j.ecns.2020.04.003>
11. Klock ACT, Wallius E, Hamari J (2021) Gamification in freight transportation: extant corpus and future agenda. *Int J Phys Distrib Logist Manag* 51(7):685–710. <https://doi.org/10.1108/IJPDLM-04-2020-0103>
12. Scharkow M, Festl R, Vogelgesang J, Quandt T (2015) Beyond the “core-gamer”: genre preferences and gratifications in computer games. *Comput Hum Behav* 44:293–298. <https://doi.org/10.1016/j.chb.2014.11.020>
13. NVIDIA (2021) NVIDIA Omniverse. Available at: <https://developer.nvidia.com/nvidia-omniverse-platform> (Accessed: 15 Sept 2021)
14. Microsoft (2021) Microsoft azure digital twin. Available at: <https://azure.microsoft.com/en-us/services/digital-twins/> (Accessed 15 Sept 2021)
15. Rouse WB, Bo KR (2005) *Organizational simulation*. Wiley
16. Costanza R, Chichakly K, Dale V, Farber S, Finnigan D, Grigg K, Heckbert S, Kubiszewski I, Lee H, Li S, Magnuszewski P, Maynard S, McDonald N, Mills R, Ogilvy S, Pert PL, Renz J, Wainger L, Young M, Ziegler CR (2014) Simulation games that integrate research, entertainment, and learning around ecosystem services. *Ecosyst Serv* 10:195–201. <https://doi.org/10.1016/j.ecoser.2014.10.001>
17. Padilla JJ, Lynch CJ, Diallo SY, Gore RJ, Barraco A, Kavak H, Jenkins B (2016) Using simulation games for teaching and learning discrete-event simulation. In: *Proceedings of the 2016 winter simulation conference (WSC)*, pp 3375–3384. <https://doi.org/10.1109/WSC.2016.7822368>
18. Baker A, Navarro EO, van der Hoek A (2005) An experimental card game for teaching software engineering processes. *J Syst Softw* 75(1–2):3–16. <https://doi.org/10.1016/j.jss.2004.02.033>
19. Kohwalter TC, Clua EWG, Murta LGP (2011) SDM—an educational game for software engineering. In: *Proceedings of the 2011 Brazilian symposium on games and digital entertainment (SBGames)*, pp 222–231. <https://doi.org/10.1109/SBGAMES.2011.10>
20. Lui RWC, Au CH (2017) Designing simulation games for information systems education—A case study in teaching for digital marketing. In: *Proceedings of the 2017 IEEE 6th international conference on teaching, assessment, and learning for engineering (TALE)*, pp 290–295. <https://doi.org/10.1109/TALE.2017.8252349>
21. Sitzmann T (2011) A meta-analytic examination of the instructional effectiveness of computer-based simulation games. *Pers Psychol* 64(2):489–528. <https://doi.org/10.1111/j.1744-6570.2011.01190.x>
22. Squire K, Klopfer E (2007) Augmented reality simulations on handheld computers. *J Learn Sci* 16(3):371–413. <https://doi.org/10.1080/10508400701413435>
23. Ruben BD (1999) Simulations, games, and experience-based learning: the quest for a new paradigm for teaching and learning. *Simul Gaming* 30(4):498–505. <https://doi.org/10.1177/104687819903000409>
24. Faria AJ, Hutchinson D, Wellington WJ, Gold S Developments in business gaming: a review of the past 40 years. *Simul Gaming* 40(4):464–487. <https://doi.org/10.1177/1046878108327585>
25. Pohl CQ, Geiser C, Lehmann W (2006) The relationship between computer-game preference, gender, and mental-rotation ability. *Pers Individ Differ* 40(3):609–619. <https://doi.org/10.1016/j.paid.2005.07.015>
26. Faria AJ, Wellington WJ (2004) A survey of simulation game users, former-users, and never-users. *Simul Gaming* 35(2):178–207. <https://doi.org/10.1177/1046878104263543>
27. Pasin F, Giroux H (2011) The impact of a simulation game on operations management education. *Comput Educ* 57(1):1240–1254. <https://doi.org/10.1016/j.compedu.2010.12.006>

28. Deshpande AA, Huang SH (2011) Simulation games in engineering education: a state-of-the-art review. *Comput Appl Eng Educ* 19(3):399–410. <https://doi.org/10.1002/cae.20323>
29. Van den Bossche P, Gijsselaers W, Segers M, Woltjer G, Kirschner P (2011) Team learning: building shared mental models. *Instr Sci* 39:283–301. <https://doi.org/10.1007/s11251-010-9128-3>
30. Steinwachs B (1992) How to facilitate a debriefing. *Simul Gaming* 23(2):186–195. <https://doi.org/10.1177/1046878192232006>
31. Boschert S, Heinrich C, Rosen R (2018) Next generation digital twin. Proceedings of the 12th international symposium on tools and methods of competitive engineering (TMCE), pp 209–218
32. DeSmet A, Shegog R, Van Ryckeghem D, Crombez G, De Bourdeaudhuij I (2015) A systematic review and meta-analysis of interventions for sexual health promotion involving serious digital games. *Games Health J* 4(2):78–90. <https://doi.org/10.1089/g4h.2014.0110>
33. Brown A, Barnes C, Byaruhanga J, McLaughlin M, Hodder RK, Booth D, Nathan N, Sutherland R, Wolfenden L (2020) Effectiveness of technology-enabled knowledge translation strategies in improving the use of research in public health: systematic review. *J Med Internet Res* 22(7):e17274. <https://doi.org/10.2196/17274>
34. Bruce B, McGrath P (2005) Group interventions for the prevention of injuries in young children: a systematic review. *Injury Prev* 11(3):143–147. <https://doi.org/10.1136/ip.2004.007971>
35. Majeed A, Baadel S, Williams ML (2017) Semantics of intelligent learning environment (ILE): cesim simulation. Proceedings of the 8th international conference on e-education, e-business, e-management and e-learning (IC4E '17), Association for Computing Machinery, pp 24–29. <https://doi.org/10.1145/3026480.3026488>
36. Zea E, Valez-Balderas M, Uribe-Quevedo A (2021) Serious games and multiple intelligences for customized learning: a discussion. In: Recent advances in technologies for inclusive well-being (Brooks AL, Brahman S, Kapralos B, Nakajima A, Tyerman J, Jain LC). *Intelligent systems reference library*, vol 196, pp. 177–189, Springer. https://doi.org/10.1007/978-3-030-59608-8_9
37. Ferreira CP, González-González CS, Adamatti DF (2021) Business simulation games analysis supported by human-computer interfaces: a systematic review. *Sensors* 21(14):4810. <https://doi.org/10.3390/s21144810>
38. Moreno R, Mayer RE (2004) Personalized messages that promote science learning in virtual environments. *J Educ Psychol* 96(1):165–173. <https://doi.org/10.1037/0022-0663.96.1.165>
39. Liu C, Cheng Y, Huang C (2011) The effect of simulation games on the learning of computational problem solving. *Comput Educ* 57:1907–1918. <https://doi.org/10.1016/j.compedu.2011.04.002>
40. Pando-García J, Periañez-Cañadillas I, Charterina J (2016) Business simulation games with and without supervision: an analysis based on the TAM model. *J Bus Res* 69(5):1731–1736. <https://doi.org/10.1016/j.jbusres.2015.10.046>
41. Kiili K (2007) Foundation for problem-based gaming. *Brit J Educ Technol* 38(3):394–404. <https://doi.org/10.1111/j.1467-8535.2007.00704.x>
42. Kanawattanachai P, Yoo Y (2007) The impact of knowledge coordination on virtual team performance over time. *MIS Q* 31(4):783–808. <https://doi.org/10.2307/25148820>
43. Kanawattanachai P, Yoo Y (2001) Dynamic nature of trust in virtual teams. *J Strateg Inf Syst* 11(3–4):187–213. [https://doi.org/10.1016/S0963-8687\(02\)00019-7](https://doi.org/10.1016/S0963-8687(02)00019-7)
44. Rulke DL, Galaskiewicz J (2000) Distribution of knowledge, group network structure, and group performance. *Manage Sci* 46(5):612–625. <https://doi.org/10.1287/mnsc.46.5.612.12>
45. D'Artista BR, Hellweger FL (2015) Urban hydrology in a computer game? *Environ Model Softw* 22(11):1679–1684. <https://doi.org/10.1016/j.envsoft.2006.09.004>
46. Tan B, Aziz AR, Chua K, Teh KC (2002) Aerobic demands of the dance simulation game. *Int J Sports Med* 23(2):125–129. <https://doi.org/10.1055/s-2002-20132>



Margaret L. Loper , Tuncer Ören , Cláudio Gomes ,
Valdemar Vicente Graciano Neto , and Ernest H. Page

Abstract

Providing the appropriate infrastructure for simulation is the topic of this chapter of the SCS M&S Body of Knowledge. It provides sections on various simulation standards, standard organizations, and compliance certificates. Publications of some applicable codes of best practices and lessons learned are provided as well as a section on resource repositories. The currently dominant standard Distributed Interactive Simulation (DIS) and High-Level Architecture (HLA) conclude the chapter.

Keywords

Modeling and simulation · Simulation standards · Simulation standard organizations · Distributed interactive simulation (DIS) · High-level architecture (HLA)

M. L. Loper (✉)

Georgia Tech Research Institute, Atlanta, GA, USA

e-mail: Margaret.Loper@gtri.gatech.edu

T. Ören

University of Ottawa, Ottawa, ON, Canada

e-mail: toren@uottawa.ca

C. Gomes

Aarhus University, Aarhus, Denmark

e-mail: claudio.gomes@ece.au.dk

V. V. Graciano Neto

Federal University of Goiás, Goiania, Brazil

E. H. Page

The MITRE Corporation, McLean, VA, USA

e-mail: epage@mitre.org

6.1 Standards

Margaret L. Loper

Standardization involves the use of common products, processes, procedures, and policies to facilitate attainment of business objectives. Standardization is about enabling interoperability: a fundamental objective of all stakeholders, be they policy-makers, industrial players, or users. Numerous industrial initiatives in a variety of different economic sectors owe their success to a commitment of the stakeholders to join forces to agree on open specifications for interoperable systems. Since the earliest days of distributed simulation, standards have played a crucial role in achieving interoperability.

The most widely used distributed simulation standards in place today are the Distributed Interactive Simulation (DIS) Protocol, the High Level Architecture (HLA), and the Test- and Training Enabling Architecture (TENA). There are various means to establish standards, and the communities responsible for these Live, Virtual, and Constructive (LVC) simulation standards have chosen different approaches.

6.1.1 De Jure, De Facto, and Proprietary Standards

There are three basic types of standards in existence today and prevalent in the IT industry:

- De Jure standard: endorsed by a standards organization (TechEncyclopedia <http://www.techweb.com/encyclopedia/defineterm.jhtml?term=de+jure+standard>);
- De Facto standard: widely used, but not endorsed by a standards organization (TechEncyclopedia <http://www.techweb.com/encyclopedia/defineterm.jhtml?term=defactostandard>); or
- Proprietary standard: belongs to an entity that exercises control over the standard.

The three types of standards are not orthogonal. There are cases where the lines between the types of standards may become blurred or combined. An example of blurring the lines between De Facto and proprietary standards is the two “standards” for High Definition DVD formats. Each standard is supported by a group of vendors, and the formats are incompatible. The expectation is that one of the proprietary standards will become the community De Facto standard for digital video recording, much like the battle some years ago between VHS and BETA formats. An example of combining types of standards is the BMD benchmark environment used by the Missile Defense Agency (MDA). The MDA simulation community has created an environment for its developers to benchmark new

algorithms and components. The environment, considered an MDA standard, is based on the proprietary MATLAB environment. Thus, MDA has created De Facto standards which use proprietary standards as its foundation.

6.1.2 Open Standards

Open standard is another term often used when discussing standards. An open standard is more than just a specification; the *principles* behind the standard and the *practice* of offering and operating the standard are what makes the standard Open. The term “open standard” may be seen from perspectives of its stakeholders (“Open Standards Requirements”, Ken Krechmer, <http://www.csrstds.com/openstds.pdf>:

- Organizations representing the standards creators consider a standard to be open if the creation of the standard follows the tenets of open meeting, consensus, and due process.
- An *implementer* of a standard would call the standard open when it serves the market they wish, it is without cost to them, does not preclude further innovation (by them), does not obsolete their prior implementations, and does not favor a competitor.
- The *user* of an implementation of the standard would call a standard open when multiple implementations of the standard from different sources are available, when the implementation functions in all locations needed, when the implementation is supported over the user-planned service life, and when new implementations desired by the user are backward compatible to previously purchased implementations.

There are numerous definitions of an open standard by national standards bodies (http://en.wikipedia.org/wiki/Open_standard). The definition by Krechmer lists ten requirements that enable open standards.

6.1.3 Standards Organizations

A standards organization is any entity whose primary activity is developing, coordinating, promulgating, revising, amending, reissuing, interpreting, or otherwise maintaining standards that address the interests of a wide base of users. There are two general types of standards organizations: standards developing organizations (SDO) and standards setting organizations (SSO).

SDOs are formal organizations accredited to develop standards using open and transparent processes. Examples include the International Organization for Standardization (ISO) and the Institute of Electrical and Electronics Engineers (IEEE). SSOs refer to organizations that set what the market perceives as standards. The term “recognized SSO” refers to any SSO recognized directly or indirectly by a

government entity. Consortia is the term used for SSOs that are not recognized SSOs. Examples of a “recognized SSO” include the World Wide Web Consortium (W3C) and the Internet Engineering Task Force (IETF).

6.1.4 M&S Standards Organizations

M&S standards organizations can be classified into two types: government and commercial. Government refers to standards forums under US government control. These types of standards organizations are typically composed of systems engineers and technical leads of major DoD stakeholders of the architecture. They discuss requirements, design trade-offs, and issues associated with the architecture. These standards organizations also have contractor support that is responsible for architecture design and prototyping. Simulation-related standards that have been created using this approach include TENA.

Commercial refers to standards created in open forums outside of government control. Examples of this include IEEE, SISO, International Organization for Standardization (ISO), and Object Management Group (OMG). These types of organizations are composed of users, vendors, academics, government organizations, and developers of the architecture. Like government forums, they discuss requirements, trade-offs, and other issues associated with the architecture. However, they do not have contractor support for architecture design and prototyping. Instead, these forums rely on members to develop prototypes and provide technical feedback on the architecture specifications.

Another model of standards development that has been successfully used for LVC architectures is a combination of government and commercial organizations. This was demonstrated with the first set of HLA standards. The government was responsible for developing and evolving the early versions of the HLA specifications. This enabled DoD stakeholders to include requirements and provide technical feedback resulting from their programs. Once they reached a point of maturity, the HLA specifications were transferred to SISO and went through IEEE standardization. The HLA standards were also taken to OMG to be standardized. Similarly, the Synthetic Environment Data Representation and Interchange Specification (SEDRIS) (<http://www.sedris.org/> and <http://en.wikipedia.org/wiki/Sedris>) standards were initially developed as government standards and then taken to ISO for standardization. Using IEEE, OMG, and ISO enabled the standards to receive a broader commercial review. Simulation-related standards that have been created using this approach include DIS, HLA, and SEDRIS.

There are two main standards developing organizations in the LVC community today: the Architecture Management Team, which develops TENA standards, and SISO, which develops DIS and HLA standards. In addition to these standards organizations, the DoD services each have a group responsible for coordinating standards use, both from developing object model content (i.e., FOMs) as well as endorsing standards that meet the requirements of their programs. These groups

have people that participate in the AMT or SISO, but they do not have formal representation nor formal requirements generation functions for these standards developing bodies.

There are also commercial standards organizations involved in developing specifications and standards for technologies related to LVC. For example, the Internet Engineering Task Force (IETF) develops communication standards, including security; the World Wide Web Consortium (W3C) develops web-related standards such as SOAP and XML; the OMG develops modeling standards such as UML, SysML, and UPDM; OASIS and the Open Group have developed specifications for the service-oriented architecture; and ISO has standardized SEDRIS. Thus, there is a hybrid approach to standards, encompassing standards and technologies from all IT-related organizations. However, there is little, if any, coordination among these standards development activities resulting in a stovepipe approach to standards management.

6.1.5 Compliance Certification

The overarching purpose of compliance certification to a standard is to ensure that products adhere to that particular standard. Compliance certification provides a level of security to users of compliant products and provides a level of assurance that certified products satisfy some set of requirements. Compliance certification is an important element of the standards process.

Compliance certification may be defined as the act or process of determining compliance to a defined standard. The primary reasons for standards compliance in the M&S LVC domain are a greater probability of interoperability between simulation assets and a greater probability for reuse of those assets in different configurations. A number of processes are in use today with existing LVC standards. Those processes range from very informal approaches such as checklists to formal compliance tests. Operational certification is most often associated with verification and validation however.

6.2 Code of Best Practice

Tuncer Ören.

“The set of best practices recommended for use for any MS&A application includes:

1. conceptual modeling practice,
2. innovative approaches,
3. software engineering practice,
4. model confidence/ verification, validation, and accreditation (VV&A),
5. use of standards,

6. interoperability,
7. execution performance, and
8. user-friendliness and accessibility” [1].
9. country modeling [2],
10. military and business [3, 4],
11. networks [5],
12. participatory modeling [6], and
13. railways [7].

Code of best practices for: Crash modeling and simulation [8], engineering applications [9, 10], healthcare [11, 12], homeland security applications [1], and modeling and simulation [13, 14].

6.3 Lessons Learned

Tuncer Ören.

The following list comprises seminal papers comprising lessons learned from selected application domains.

6.4 Resource Repositories

Valdemar V. Graciano Neto and Cláudio Gomes.

The multiplicity of Modeling and Simulation (M&S) formalisms and simulation paradigms is high. That diversity often forces researchers to produce their own simulation models from scratch every time they initiate a new project due to difficulties in reusing existing models. Those difficulties range between (i) not knowing whether similar models already exist, (ii) differences in formalisms, even when models were produced for a similar domain, and (iii) lack of documentation about how to use such models. The Modeling and Simulation Resources Repositories (MSRR), also known as resource libraries or suites, have potential to foster reuse by gathering a diversity of resources in a unified access point. Resources (also known as assets or artifacts) such as models (simulatable or not-simulatable [15]), experimental frames, pairs of base and lumped models [16], specifications of physical environments and scenarios, datasets, composable simulation components and simulation services can be made available to a large audience [17]. Models capturing specific domains can be available in several formats, such as XML, UML, or DEVS, and model transformations can also be available to transform non-executable models into simulatable formats.

Sharing and exchanging models have the potential to accelerate the systems development. Efforts have been made, for instance, to standardize the representation of physical system models, through languages such as Modelica [18], and interfaces

such as the Functional Mockup Interface (FMI) standard [19]. Modelica standard library is a collection of modular physical system models and common block diagram elements enabled by the Modelica Language, while FMU Cross Check Repository is a collection of black box simulators exported from different modeling and simulation tools. The Modelica standard library allows a researcher to quickly create a model for a physical system by reusing pre-existing components in the standard library. The FMI standard, in turn, through its black box and Intellectual Property (IP) protecting interface, enables an unprecedented level of integration of models (as black boxes) provided by different and even competing suppliers.

These advances democratize M&S by making it cheaper to produce high-quality models of the system, which in turn can be more easily exchanged with researchers. Smaller companies and universities can then reap the benefits of M&S, speeding up innovation. Another advantage brought by resources repositories is that they enable benchmarking. For instance, researchers who create a new machine learning technique can apply it to many freely available datasets. Over time, benchmarks emerge when researchers are expected to tackle their contributions. This leads to more mature contributions and easy comparison with existing ones.

Ideally, a resource repository should be capable of: Catalog/index/organize the resources stored, persist, and allow for resources search and retrieve, resource management, and resources delivery through well-defined interfaces, resource stores (analogous to application stores), and as services (simulation as a service, for instance) [20]. Resources repositories are common in other areas, such as software engineering [21, 22] and biology [23]. MSRR have also been proposed over the years [24–26] (<https://ntrs.nasa.gov/citations/20060023324>).

However, several challenges still remain, such as a standardized representation in order to enable their existence. This is hard to be achieved due to the diversity of formalisms, which can be categorized as [27–31]:

- **Time Domain**—The time can be a singleton (e.g., algebraic equations), a continuous set (e.g., Ordinary and Algebraic Differential Equations), discrete set (e.g., Difference equations, Petri-nets, Automata), or superdense set (e.g., Hybrid Automata and Classic DEVS). In Superdense time [32–34], each time point is a pair consisting of a real number and a natural.
- **State Domain**—The state domain can be a continuous set (e.g., ODEs and DAEs), a discrete set (e.g., Petri-nets), or a mix of both (e.g., DEVS and Hybrid Automata).
- **Behavior Trace**—The behavior trace can be discontinuous (e.g., DEVS and Hybrid Automata), and continuous (e.g., ODEs and DAEs).
- **Causality**—Models can be a-causal, when they can be coupled to other models without any notion of inputs and outputs (e.g., DAEs), or causal, when outputs need to be connected to inputs and vice-versa (e.g., DEVS and Difference Equations).

- Evolution—The evolution of the state can be deterministic (e.g., DEVS), stochastic (e.g., Markov Chains), or non-deterministic (e.g., Hybrid Automata). To overcome these difficulties, the following suggestions have been made in the literature to conform different formalisms to make them be coupled [35]:
- Super Formalism—The formalisms used to express each model are unified into one single formalism, with well-defined syntax and semantics. This is what is done in [36, 37]. Other examples include: timed Petri-nets, Markov chains, etc.
- Common Formalism Reduction—The models are transformed into a model that is expressed in a single formalism. The “common” adjective refers to the fact that each model can be transformed into a restricted set of formalisms. Hence, one formalism must be found to which all models to be integrated can be transformed. For example, differential equations can be used to represent the model of a PID controller sampling a plant model. The latter was originally modeled as a differential equation. More examples are detailed in [35].

Co-simulation can be seen as taking the common formalism reduction integration technique to the extreme, where models that produce behavior are coupled solely on their behavior (inputs/outputs, over time). However, automatically configuring co-simulation can be very difficult [29, 38].

For more challenges and potential solutions to establishing model repositories, we recommend the following references. Basciani et al. [23] established a discussion on the reality of resource repositories some years ago. Zeigler et al. [20] show how to build a model suite relying on the MS4 Me tool and Ören [17] presents requirements necessary to achieve reuse through MSRR.

6.5 Distributed Interactive Simulation (DIS)

Ernest H. Page, Margaret L. Loper.

For nearly a half-century, the defense simulation community has explored, developed, and applied technologies and methods that support the runtime interoperation of simulations and other systems. Major milestones in this history are:

- DARPA Simulator Networking (SIMNET) program [39–42]
- Distributed Interactive Simulation (DIS) protocol [43–47]
- Aggregate Level Simulation Protocol (ALSP) [48, 49]
- High Level Architecture (HLA) for Modeling and Simulation [50], and
- Test and Training Enabling Architecture (TENA) [51]

Within this M&SBoK, we highlight DIS and HLA. But a brief discussion of SIMNET is warranted to establish the context for each of these distributed simulation standards.

6.5.1 Simnet

Shortly after the development of the ARPANET, DARPA initiated the SIMNET program to investigate the feasibility of using networked simulators to support group training (also referred to as *collective* training) at large scales and at great distances. The SIMNET vision was a large-scale, interactive, collection of networked simulations forming a *synthetic environment* that could be entered by any authorized combatant from anywhere on the network using his/her simulator as a porting device. The initial project scope called for a simulator networking testbed with four geographically distributed sites hosting 50–100 vehicle simulators each, with a focus on slower-moving ground-based platforms, e.g., tanks and armored personnel carriers. The project required technological advances in a variety of areas, including image generation, distributed databases, and real-time network protocols. Key design principles for SIMNET included:

- *Selective fidelity.* In order to minimize simulator costs, a simulator should only contain high fidelity representations of those elements essential to the training task. Everything else should be represented at lower fidelities, or not all.
- *Autonomous simulation nodes.* Each node is responsible for maintaining the state of at least one object in the synthetic environment, and for communicating to other nodes any events caused by its object(s). Each node receives event reports from other nodes and calculates the effects of those events on its objects. All events are broadcast on the simulation network and are available to any node that is interested. There is no centralized controlling process. Nodes may join and leave the network without affecting other nodes. Each node advances simulation time according to a local clock.
- *Transmission of ground truth data.* Each node transmits the absolute truth about the current state of the object(s) it represents. Alteration of data to suit simulation objectives is the responsibility of the receiving node. For example, the position of a vehicle is broadcast to the network with 100% accuracy. If an object in another simulator determines that it would perceive the vehicle through a particular sensor, with an accuracy determined by the alignment of the sensor and current weather conditions, then the receiving simulator should degrade the reported position accordingly.
- *Transmission of state change information.* To minimize network loading, nodes transmit state update information only. To accommodate late-joining nodes and networks with high packet loss, this rule is often relaxed. In these situations, nodes send periodic (but relatively infrequent) updates for each owned object regardless of whether or not their state changes. This update interval is known as the “heartbeat.”
- *Dead reckoning.* Between state update messages, receiving nodes may extrapolate the last reported state of remote objects that are of interest. To keep the extrapolated values and actual values roughly aligned, the sending node maintains the same approximation used by the receiving node(s) and transmits a state

update whenever the true position (or orientation) of an object diverges from the calculated dead reckoned values by more than an agreed-upon threshold. Lin [52] and Fujimoto [53] discuss common dead reckoning algorithms.

SIMNET was adopted by the Army as the basis for the Combined Arms Tactical Trainer (CATT) in 1990 and continued to be used in a variety of programs until supplanted by the DIS standard. SIMNET has been identified as one of the most significant transitions of technology from DARPA to DoD [40].

6.5.2 Origins of the DIS Protocol

Recognizing the importance of the SIMNET program and concerned that activity related to networked simulation was occurring in isolation, a small conference was held in April 1989 called “Interactive Networked Simulation for Training.” The group believed that if there were a means to exchange information between companies, distributed simulation technology would advance more rapidly. The group also believed that technology had stabilized enough to begin standardization. The conference developed into the Distributed Interactive Simulation (DIS) Workshops.

Through these workshops, networked simulation technology and the consensus of the community were captured in proceedings and standards. The standards initially focused on SIMNET, but evolved to include a broader range of technology areas. DIS Workshops were held semi-annually from 1989 through 1996. In 1996, the DIS Workshops transformed itself into a more functional organization called the Simulation Interoperability Standards Organization (SISO), which focused on creating standards for the broad area of simulation interoperability. The first Simulation Interoperability Workshop (SIW) held under the SISO banner was the 1997 Spring SIW in Orlando. SIWs have continued since 1997, holding some workshops at various locations in Europe.

The Distributed Interactive Simulation (DIS) protocols became the Institute of Electrical and Electronics Engineers (IEEE)1278.1 standard in 1993. The fundamental design principles for DIS follow directly from SIMNET, and much of the standardization effort focused on extending the basic SIMNET communication structure—the Protocol Data Unit (PDU)—a bit-encoded packet for communicating entity state and other types of information necessary for distributed combat simulations, e.g., weapons fire and weapons detonation events.

Like SIMNET, DIS was designed to support the internetworking of simulations that run in real-time. Whereas SIMNET had achieved the ability to support relatively small numbers of concurrently running simulators representing platoon and squad-sized engagements, the vision for DIS was to support the interoperation of thousands of simulators/simulations and scale to a military campaign level (tens to hundreds of thousands of battlefield entities). This appetite for scale led to a burgeoning market in Semi-Automated Forces (SAF). SAFs—a concept initiated within SIMNET—were used to populate synthetic environments with background objects that behaved in a “reasonable” way [46]. They were dubbed

“semi-automated” because human intervention was often required to make the modeled entities maintain their reasonable behavior. However, the power and utility of SAFs were recognized very quickly. Entity behavior in SAFs became the focus of numerous conferences, workshops, and texts. SAFs were a ripe area for research in Artificial Intelligence engines such as Soar [54]. DIS-supported simulation environments consisting entirely of SAFs became commonplace.

One of the lasting contributions of the DIS Workshops was the definition of Live, Virtual, and Constructive (LVC) simulations. This taxonomy categorizes simulations by the way in which humans interact with them. Live simulation refers to real people operating real systems (e.g., a pilot flying a jet) for a simulated purpose. A virtual simulation is one that involves real people operating simulated systems (e.g., a pilot flying a simulated jet). Constructive simulations are those that involve simulated people operating simulated systems (e.g., a simulated pilot flying a simulated jet).

6.5.3 DIS Today

The goal of DIS is to create a common, consistent simulated world where different types of simulators can interact. Central to achieving this goal is a set of IEEE standards. The most commonly used standard is 1278.1, which describes the PDUs. The first DIS standard defined 10 PDUs; the most recent standard, DIS 7, was published in 2012 and defines 72 PDUs arranged into 13 families. The approved IEEE Standards for DIS include:

- IEEE 1278.1—Application Protocols
- IEEE 1278.1A—Enumeration and Bit-encoded Values
- IEEE 1278.2—Communication Services and Profiles
- IEEE 1278.3—Exercise Management & Feedback (EMF)
- IEEE 1278.4—Verification Validation and Accreditation
- IEEE P1278.5—XXXX—Fidelity Description Requirements (never published).

In addition to the IEEE standards, SISO maintains and publishes an “enumerations and bit-encoded fields” document yearly. This document is referenced by the IEEE standards and used by DIS, TENA, and HLA.

From an implementation perspective, simulation owners either custom-develop DIS interfaces or buy commercial products. There is also an open-source initiative, Open-DIS, to provide a full implementation of the DIS protocols in C++ and Java [55].

There have been numerous DIS federation events over the last 25 years. Two examples are “bookend” LVC events presented at the Interservice/Industry Training, Simulation and Education Conference (I/ITSEC). Twenty-three years spanned the two events, and while technology has progressed, some aspects have not progressed as quickly as we might think. The 1992 event was the first-ever

demonstration of DIS and distributed simulation among dissimilar, heterogeneous simulations [45]. The 2015 event was an effort to recreate the demonstration with modern technology and architectures [56].

6.6 High Level Architecture (HLA)

Ernest H. Page, Margaret L. Loper.

By 1995, the evidence was clear that interconnecting simulations could be of practical value. SIMNET provided an efficient and effective mechanism for linking man-in-the-loop simulators. DIS extended SIMNET and provided scalability to many thousands of entities in SAF-based exercises. Another DARPA project, the Aggregate Level Simulation Protocol (ALSP), developed a capability to interconnect “logical time,” e.g., discrete event, simulations [49]. Also by this time, many defense simulations had interconnection interfaces—some SIMNET, some DIS, some ALSP, some “homegrown,” and some had multiple such interfaces. To mitigate against the proliferation of interconnection approaches, the DoD, through the Defense Modeling and Simulation Office (DMSO) and SISO, began developing a standard for simulation interconnection known as the High Level Architecture (HLA). The HLA was envisioned as an approach to bridge live, virtual, and constructive simulations in one architecture, representing a generalization and extension of SIMNET, DIS, and ALSP. The HLA architecture is defined by three components:

- An Object Model Template—a common model definition and specification formalism,
- An Interface Specification—a collection of services describing the HLA runtime environment, and
- The HLA Rules—governing compliance with the architecture.

The HLA standards began in 1995 under a government standards process managed by DMSO. The DoD adopted the baseline HLA architecture in 1996 and the standards were moved to an open standards process managed by SISO. The IEEE standards for HLA, first approved in 2000 and updated in 2010, include:

- 1516—Framework and Rules
- 1516.1—Federate Interface Specification
- 1516.2—Object Model Template (OMT) Specification
- 1516.3—Federation Development and Execution Process (FEDEP) Recommended Practice
- 1516.4—Recommended Practice for Verification, Validation, and Accreditation of a FederationAn Overlay to the HLA FEDEP

The HLA was conceived to have applicability across the full range of defense simulation applications, including those used to support training, mission rehearsal, analysis, and test and evaluation.

At core of HLA is the notion of a *federation*. A federation is a collection of federates—simulations and other systems—that interoperate using the protocols described by the architecture. The HLA is based on the idea of separating the functionality of simulations from the infrastructure required for communication among simulations. This separation is accomplished by a distributed operating system called the Run-Time Infrastructure (RTI). The RTI provides common services to simulation systems and provides efficient communications to logical groups of federates. Federation execution is accomplished through the RTI, which is an implementation of the services defined by the interface specification.

In contrast to SIMNET and DIS, HLA includes time management services to support event ordering [57]. Both time stamp order, where messages are delivered to simulations in order of time stamp, and receive order, where messages are delivered to simulations in order received, are supported in HLA. While HLA provides global time management, use of these services is not required. Simulations can choose to advance time at its own pace, not synchronized with other simulations.

In contrast to the static DIS PDUs, HLA uses the concept of OMTs to specify the information communicated between simulations. This enables users to customize the types of information communicated among federates based on the needs of the federation. A Federation Object Model (FOM), and instantiation of the OMT, provides the model specification and establishes a contract between the federates with respect to the nature of the activity taking place during federation runtime.

In a typical federation execution, a federate joins the federation, indicates its operating parameters (e.g., information the federate will provide to the federation and information it will accept from the federation), and then participates in the evolution of federation state until the federate departs the federation, or the simulation terminates. FOM data is provided to the RTI at runtime, enabling the infrastructure to enforce the information contract that the FOM represents.

In 1996, HLA compliance was mandated for all defense simulations, with the intention that support for other protocols would cease [58]. To accommodate DIS applications the Real-time Platform Reference (RPR), FOM was developed which defines a translation between DIS PDUs and HLA services [59]. As with an earlier mandate of the programming language Ada, however, the “No Can Pay/No Can Play” HLA mandate was perceived as onerous and became too unwieldy to enforce.

Distributed simulation architectures are designed to meet the needs of one or more user communities, and the design choices made by the HLA attempted to improve on perceived shortcomings of existing architectures [60]. The static nature of DIS PDU’s was identified as a significant problem; as the real world is always changing. A flexible object model capable of modeling changing data without having to continuously change the underlying standard was seen as a better approach. Allowing users to define their data exchange based on specific requirements using the OMT was seen as providing improved object model extensibility.

However, increased flexibility to the user also allowed users to independently develop a plethora of object models that were rarely interoperable. Additionally, HLA adopted an API Standard as opposed to an on-the-wire standard that allowed it to more rapidly adopt technological advancements in how data are transmitted. While this enabled commercial RTI developers the freedom to innovate and optimize their RTI implementations, the result was non-interoperable RTIs. In practice, when disparate RTI versions are used in a given event, gateways or other inter-protocol translation mechanisms are used to bridge the federates.

Today, both HLA-compliant and DIS-compliant simulations abound. Since HLA separates the functionality of simulations from the infrastructure, it has had more success in being adopted by non-DoD applications, including NASA, transportation, and supply chain management. The existence of multiple architectures means users will select the methodology that best meets their needs. This often results in multiple architectures being used in the same federation execution. In this case, incompatibilities between DIS, HLA, and TENA require the development of point solutions to effectively integrate the various architectures into a single, unified set of simulation services. The future of distributed simulation to solve and understand complex problems will rely on the development of simulation standards.

References

1. Jain S, McLean CR (2011) Best practices for modeling, simulation and analysis (MS&A) for homeland security applications. U.S. Dept. of Commerce, National Institute of Standards and Technology. NISTIR 7655. <https://www.govinfo.gov/content/pkg/GOVPUB-C13-ae22d29a1ea5577f0dc3f67b51bc9ba/pdf/GOVPUB-C13-ae22d29a1ea5577f0dc3f67b51bc9ba.pdf>
2. Silverman B et al (2021) StateSim: lessons learned from 20 years of a country modeling and simulation toolset. Social computing, behavioral-cultural modeling and prediction and behavior representation in modeling and simulation annual conference. https://repository.upenn.edu/cgi/viewcontent.cgi?article=1939&context=ese_papers
3. Barth R et al (2021) typical pitfalls of simulation modeling - lessons learned from armed forces and business. *J Artif Soc Soc Simul* 15(2). <https://doi.org/10.18564/jass.1935>
4. Hofmann MA (2004) Criteria for decomposing systems into components in modeling and simulation: lessons learned with military. <https://doi.org/10.1177/0037549704049876>
5. Anton SD (2018) The dos and don'ts of industrial network simulation: a field report. ISCSIC '18: proceedings of the 2nd international symposium on computer science and intelligent control, September 2018 Article No.: 6, pp 1–8 <https://doi.org/10.1145/3284557.3284716>
6. Sterling EJ et al (2019) Try, try again: Lessons learned from success and failure in participatory modeling. <https://doi.org/10.1525/elementa.347>
7. Meijer S (2012) Gaming simulations for railways: lessons learned from modeling six games for the dutch infrastructure management. <https://doi.org/10.5772/35864> <https://www.intechopen.com/books/infrastructure-design-signalling-and-security-in-railway/gaming-simulations-for-railways-lessons-learned-from-modeling-six-games-for-the-dutch-infrastructure>
8. Fasanella EL et al (2002) Best practices for crash modeling and simulation. NASA/TM-2002–211944, ARL-TR-2849. <https://ntrs.nasa.gov/api/citations/20020085101/downloads/20020085101.pdf>
9. Fasanella EL, Jackson KE (2002) Best practices for crash modeling and Simulation. NASA/TM-2002–211944, ARL-TR-2849. https://www.researchgate.net/publication/24302082_Best_Practices_for_Crash_Modeling_and_Simulation

10. Mare JC (2019) Best practices for model-based and simulation-aided engineering of power transmission and motion control systems. *Chin J Aeronaut* 32(1):186–199. <https://www.sciencedirect.com/science/article/pii/S1000936118302668>
11. Erdemir A et al (2020) Credible practice of modeling and simulation in healthcare: ten rules from a multidisciplinary perspective. *J Transl Med* 18:369. <https://translational-medicine.biomedcentral.com/track/pdf/https://doi.org/10.1186/s12967-020-02540-4.pdf>
12. Standards of Best Practices: Simulation. INACSL (The International Nursing Association for Clinical Simulation and Learning). <https://www.inacsl.org/inacsl-standards-of-best-practice-simulation/>
13. JHU-APL (2010). Best Practices for the Development of Models and Simulations. Final Report, NSAD-L-2010–129, JNC04. <https://acqnotes.com/Attachments/JHU%20APL%20Best%20Practices%20for%20the%20Development%20of%20M&A.%20June%202010.pdf>
14. Morse KL (2010) Models and simulations development best practices. In: 13th annual NDIA systems engineering conference. https://ndiastorage.blob.core.usgovcloudapi.net/ndia/2010/systemengr/WednesdayTrack5_10735Morse.pdf
15. Zeigler B, Sarjoughian H (2017) Model development and execution process with repositories, validation, and verification. In: *Guide to modeling and simulation of systems of systems*, Springer
16. Zeigler BP (2019) *Simulation-based evaluation of morphisms for model library organization*. In: *Model engineering for simulation*, Academic Press
17. Ören T (2005) Toward the body of knowledge of modeling and simulation. In: *Interservice/industry training, simulation, and education conference (I/ITSEC)*, NTSA
18. Tiller M (2012) *Introduction to physical modeling with modelica*, vol 615. Springer
19. Blochwitz T et al (2012) Functional mockup interface 2.0: the standard for tool independent exchange of simulation models. In: *Proceedings of the 9th international modelica conference*. Linköping University Electronic Press.
20. Zeigler BP et al (2013) Creating suites of models with system entity structure: global warming example. In: Wainer GA, Mosterman PJ, Barros FJ, Zacharewicz G (eds) *Proceedings of the spring simulation conference*, IEEE
21. France RB, Bieman JM, Mandalaparty SP, Cheng, Betty HC, Jensen AC (2012) Repository for model driven development (ReMoDD). In: Glinz M, Murphy GC, Pezzè M (eds) *Proceedings of the international conference on software engineering*, IEEE.
22. Koegel M, Helming J (2010) EMFStore: a model repository for EMF models. In: Kramer J, Bishop J, Devanbu PT, Uchitel S (eds) *Proceedings of the 32nd ACM/IEEE international conference on software engineering*, vol 2. IEEE.
23. Basciani F et al. (2015) Model repositories: Will they become reality? *Proceedings of the International Workshop on Model-Driven Engineering on and for the Cloud (CloudMDE)*, Richard Paige, Jordi Cabot, Marco Brambilla, James H. Hill (Eds.), CEUR.
24. Liu HB, Su HY, Chai XD, Zhang H, Zhan SY (2009) Research and application of XML in modeling and simulation resource repository. *J Syst Simul* 22
25. Milroy A, Hale J (2006) *Modeling and simulation resource repository (MSRR) (System Engineering/integrated M&S Management Approach)*, Audrey Milroy, QTEC, Inc. and Joe Hale, NASA MSFC
26. Yan B, Yang Z (2009) Simulation resource selection based on local optimization method in distributed simulation repository. In: *Proceedings of the 16th international conference on industrial engineering and engineering management*, IEEE
27. Gomes C, Thule C, Broman D, Larsen PG, Vangheluwe H (2017) Co-simulation: state of the art. University of Antwerp, February 1
28. Gomes C, Thule C, Broman D, Larsen PG, Vangheluwe H (2018) Co-simulation: a survey. *ACM Comput Surv* 51(3):49:1–49:33
29. Gomes C (2019) Property preservation in co-simulation. University of Antwerp. <https://repository.uantwerpen.be/docman/irua/57f437/163840.pdf>

30. Jantsch A, Sander I (2005) Models of computation and languages for embedded system design. *IEEE Proc Comput Digital Tech* 152(2), 15: 114–129, IEEE
31. Vangheluwe H (2008) Foundations of modelling and simulation of complex systems. *Electronic Communications of the EASST* 10
32. Lee EA, Zheng H (2005) Operational semantics of hybrid systems. In: Morari M, Thiele L (eds) *Hybrid systems: computation and control*. Lecture Notes in Computer Science, vol 3414. Springer, pp 25–53
33. Maler O, Manna Z, Pnueli A (1992) From timed to hybrid systems. In: de Bakker JW, Huizing C, de Roever WP, Rozenberg G (eds) *Real-time: theory in practice*, lecture notes in computer science. Springer
34. Manna Z, Pnueli A (1993) Verifying hybrid systems. In: Clarke E, Henzinger T, Veith H, Bloem R (eds) *Handbook of model checking*. Lecture Notes in Computer Science. Springer
35. Vangheluwe H, De Lara J, Mosterman PJ (2002) An introduction to multi-paradigm modelling and simulation. In: Zeigler B, Barros F (eds) *Proceedings of the AI, simulation and planning in high autonomy systems conference*, IEEE
36. Mustafiz S et al (2021b) Towards modular language design using language fragments: the hybrid systems case study. In: Latifi S (ed) *Proceedings of the 13th international conference on information technology—new generations (ITNG)*, Springer
37. Mustafiz S et al (2016a) Modular design of hybrid languages by explicit modeling of semantic adaptation. In: Barros F, Prähofer H, Hu X, Denil J (eds) *Proceedings of the symposium on theory of modeling and simulation: DEVS integrative M&S symposium*, SCS
38. Oakes BJ, Gomes C, Holzinger FR, Benedikt M, Denil J, Vangheluwe H (2021) Hint-based configuration of co-simulations with algebraic loops. In: De Rango F, Wagner G, Werner F, Ören T (eds) *Simulation and modeling methodologies, technologies and applications, advances in intelligent systems and computing*, Springer
39. Cosby L (1995) SIMET: an insider's perspective. IDA Document D-1661. Institute for Defense Analyses, Washington, D.C
40. Miller DC (2015) SIMNET and beyond: a history of the development of distributed simulation. In: *Proceedings of the interservice/industry training, simulation, and education conference*. Orlando, FL
41. Pope AR (1991) The SIMNET network and protocols. Report No. 7627, BBN Systems and Technologies, Cambridge, MA
42. Thorpe J (2010) Trends in modeling, simulation and gaming: personal observations about the past thirty years and speculation about the next ten. In: *Proceedings of the interservice/industry training, simulation, and education conference*. Orlando, FL
43. DIS Steering Committee (1994) The DIS vision, a map to the future of distributed simulation. Institute for Simulation and Training
44. Hofer RC, Loper ML (1995) DIS today [Distributed interactive simulation]. *Proc IEEE* 83(8): 1124–1137
45. Loper M, Goldiez B, Smith S (1993) The 1992 IITSEC distributed interactive simulation interoperability demonstration. In: *Proceedings of the interservice/industry training, simulation, and education conference*. Orlando, FL
46. Shiflett JE (2013) Observations on the development and implementation of distributed simulation. In: *Proceedings of the interservice/industry training, simulation, and education conference*. Orlando, FL
47. Voss LD (1993) *A revolution in simulation: distributed interaction in the '90s and beyond*. Pasha Publications Inc., Virginia
48. Page EH, Canova BS, Tufarolo JA (1997) A case study of verification, validation and accreditation for advanced distributed simulation. *ACM Trans Model Comput Simul* 7(3)
49. Weatherly R, Seidel D, Weissman J (1991) Aggregate level simulation protocol. In: *Proceedings of the summer computer simulation conference*. Baltimore, MD
50. Kuhl F, Weatherly R, Dahmann J (1999) *Creating computer simulation systems: an introduction to the high level architecture*. Prentice Hall

51. Powell ET, Noseworthy JR (2014) *The test and training enabling architecture (TENA). Engineering principles of combat modeling and distributed simulation*. Wiley
52. Lin KC (1995) Dead reckoning and distributed interactive simulation. In: *Distributed Interactive simulation systems for simulation and training in the aerospace environment: a critical review*, vol 10280. International Society for Optics and Photonics, p 1028004
53. Fujimoto RM (2000) *Parallel and distributed simulation systems*. Wiley, New York
54. Jones RM, Laird JE, Nielsen PE (1996) Moving intelligent automated forces into theater-level scenarios. In: *Proceedings of the sixth conference on computer generated forces and behavioral representation*
55. McGregor D, Brutzman D (2008) Open-DIS: an open source implementation of the DIS protocol for C++ and Java. In: *proceedings of the fall simulation interoperability workshop*. Orlando, FL
56. Gritton J, Kotick D, Fraas G, Dean C (2016) Washington, we have a problem: the foundation for live virtual constructive (LVC) exercises requires fixing!. In: *Proceedings of the interservice/industry training, simulation and education conference*
57. Fujimoto RM (1998) Time management in the high level architecture. *SIMULATION* 71(6): 388–400
58. Hollenbach JW (2009) Inconsistency, neglect and confusion: a historical review of DoD distributed simulation architecture policies. In: *Proceedings of the spring simulation interoperability workshop*. Orlando, FL
59. Möller B, Dubois A, Le Leydour P, Verhage R (2014) RPR FOM 2.0: a federation object model for defense simulations. In: *Proceedings of the fall simulation interoperability workshop*. Orlando, FL
60. Henninger A, Cutts D, Loper M, Lutz R, Richbourg R, Saunders R, Swenson S (2008) *Live virtual constructive architecture roadmap (LVCAR) final report*. Modeling and simulation coordination office project no. 06OC-TR-001
61. Loper M, Cutts D (2008) *Live virtual constructive (LVC) architecture roadmap (AR) comparative analysis of standards management*. DoD Office of Security Review, Washington, DC
62. Gomes C, Oakes BJ, Moradi M, Gamiz AT, Mendo JC, Dutre S, Denil J, Vangheluwe H (2019) HintCO—hint-based configuration of co-simulations. In: Obaidat M, Ören T, Szczerbicka H (eds) *Proceedings of the international conference on simulation and modeling methodologies, technologies and applications*. Springer.
63. Calvin J, Dickens A, Gaines B, Metzger P, Miller D, Owen D (1993) The SIMNET virtual world architecture. In: *Proceedings of IEEE virtual reality annual international symposium*, IEEE, pp 450–455
64. Miller DM, Thorpe JA (1995) SIMNET: the advent of simulator networking. *Proc IEEE* 83(8):1114–1123



Reliability and Quality Assurance of M&S

7

Tuncer Ören , Valdemar Vicente Graciano Neto , Paul K. Davis ,
and Bernard P. Zeigler 

Abstract

This chapter of the SCS M&S Body of Knowledge begins with a section on the types and sources of various errors, including an extensive list of relevant terms. The need for reliability leads to a section on validation as well as a section on verification. It is closed by a section on failure avoidance.

Keywords

Modeling and simulation · sources of errors · validation · verification · failure avoidance

T. Ören (✉)
University of Ottawa, Ottawa, ON, Canada
e-mail: toren@uottawa.ca

V. V. Graciano Neto
Federal University of Goiás, Goiania, Brazil

P. K. Davis
The RAND Corporation and the Pardee RAND Graduate School,
Santa Monica, CA, USA
e-mail: pdavis@rand.org

B. P. Zeigler
University of Arizona, Arizona, USA
e-mail: zeigler@rtsync.com

7.1 Errors—Types and Sources

Tuncer Ören

7.1.1 Definitions

The term “error” has the following meanings (Merriam-Webster-error: <https://www.merriam-webster.com/dictionary/error>):

1. a. : an act or condition of ignorant or imprudent deviation from a code of behavior
- b. : an act involving an unintentional deviation from truth or accuracy
 //made an *error* in adding up the bill
- c. : an act that through ignorance, deficiency, or accident departs from or fails to achieve what should be done
 //an *error* in judgment: such as
- (1) : a defensive misplay other than a wild pitch or passed ball made by a baseball player when normal play would have resulted in an out or prevented an advance by a base runner
- (2) : the failure of a player (as in tennis) to make a successful return of a ball during play
- d. : a mistake in the proceedings of a court of record in matters of law or of fact
2. a. : the quality or state of erring
 // the map is in *error*
- b. *Christian Science*: illusion about the nature of reality that is the cause of human suffering: the contradiction of truth
- c. : an instance of false belief
3. something produced by mistake
 // a typographical *error*
 especially: a postage stamp exhibiting a consistent flaw (such as a wrong color) in its manufacture
4. a. : the difference between an observed or calculated value and a true value
 specifically: variation in measurements, calculations, or observations of a quantity due to mistakes or to uncontrollable factors
- b. : the amount of deviation from a standard or specification
5. a deficiency or imperfection in structure or function
 // an *error* of metabolism.”

7.1.2 Types of Errors

Table 7.1 lists 360 types of errors. Some types of errors are not related with M&S. An example is medical errors (that result in deaths). A recent (2021 March) blog states that: “A recent study from Johns Hopkins suggests that medical errors are now the third-leading cause of death in the U.S., having surpassed strokes, Alzheimer’s, and diabetes. In addition, *one in seven Medicare patients* receiving care in a hospital are victims of a medical error. However, medical errors can occur in almost any healthcare setting including hospitals, clinics, surgery centers, medical offices, nursing homes, pharmacies, and patients’ homes. This post will explore the most common causes of medical errors” [1].

Considering useful and sometimes even vital contributions of M&S to many disciplines [2], it would be worthwhile to explore extensive use of M&S in healthcare to lower medical errors.

7.1.3 Terms Related with Errors

Table 7.2 lists terms related with “errors”.

7.1.4 Terms (Other Than Error) Related with Failure

See Table 7.3.

7.1.5 Other Sources of Failure

There are other sources of errors in addition to those listed in Tables 7.1, 7.2 and 7.3. They are based on several types of biases such as cultural biases and dysrationalia.

The value systems of cultures differ—as clarified by Hofstede and debated by his critiques—(Wikipedia) hence cultural priorities can be different.

Dysrationalia—The inability to think and behave rationally despite adequate intelligence—developed by Stanovich [4] has important implication in decision-making.

The achievements in artificial intelligence (AI) increases expectations from it. However, it would be prudent to consider AI failures [5].

Table 7.1 List of 360 types of errors (terms are selected from the TBD informatics dictionary)

Absolute error
Acceleration error
Acceptance error
Access error
Accessibility error
Accidental error
Accounting error
Accumulated error
Accumulation error
Activation error
Active error
Adjustment error
Algorithm error
Alpha error
Ambiguity error
Analysis error
Angular error
Anticipated error
Anticipation error
Application error
Approximation error
Ascertainment error
Associative activation error
Assumption error
Asymptotic standard error
Attribute error
Attribution error
Authentication error
Authorization error
Average error
Azimuth error
Azimuthal error
Babbling error
Balance error
Balanced error
Barometric error
Benchmark error
Beta error
Bias error
Biased error
Bit error
Bus error

(continued)

Table 7.1 (continued)

Calculation error
Calibration error
Capture error
Chance error
Chaotic error
Circular error
Classification error
Clear error
Clerical error
Combined error
Command error
Communication error
Compass error
Compensated error
Compensating error
Compilation error
Compile-time error
Composite error
Computational error
Computer error
Computerization error
Conceptual error
Configuration error
Connection error
Consistency error
Constant error
Constraint error
Control error
Convergence error
Copying error
Corrected error
Correlated error
Course error
Creation error
Cultural bias error
Cultural perception error
Cumulative error
Cylindrical error
Damping error
Data error
Database error
Data-driven error

(continued)

Table 7.1 (continued)

Data-entry error
Decision error
Deductive error
Definition error
Deflection error
Deletion error
Description error
Design error
Detected error
Detection error
Device error
Digitization error
Discretization error
Disk error
Disk seek error
Disk write error
Driver error
Dumping error
Dynamic error
Encoding error
Encryption error
Environment error
Equipment error
Error of closure
Error of judgment
Error of measurement
Error of the first kind
Error of the second kind
Error of the third kind
Estimation error
Ethical error
Evidentiary error
Executable error
Execution error
Experimental error
Experimentation error
Exposure error
External error
Facial recognition error
False alarm error
Fatal error
Fencepost error

(continued)

Table 7.1 (continued)

Fiducial error
Fiducial registration error
File creation error
File error
Filing error
Formatting error
Full-scale error
Gain error
General error
Generalization error
Grammatical error
Hard error
Hardware error
Hardware installation error
Harmless error
Heeling error
Heuristic error
Human error
Hypothesis error
Hysteresis error
Identification error
Illegal access error
Illegal error
Illegal seek error
Inadvertent human error
Inconsistent accessibility error
Inconsistent formula error
Index error
Initialization error
Input quantization error
Input/output error
Inscription error
Installation error
Installer error
Instrument error
Instrumental error
Instrumentation error
Internal error
Interpolation error
Interpolation error
Interpretation error
Irrecoverable error

(continued)

Table 7.1 (continued)

Irregular error
Judgment error
Language error
Legal error
Legislative error
Lexical error
Linearization error
Literal error
Loading error
Local error
Logic error
Logical error
Loss-of-activation error
Machine error
Macro error
Mean error
Mean-square error
Measurement error
Measuring error
Measuring instrument error
Mechanical error
Medical error
Memory error
Message error
Method error
Miss error
Missed error
Misspecification error
Mode error
Model error
Modeling error
Moral error
Network device error
Network error
Network permission error
Non-sampling error
Non-spherical error
Numerical error
Obi-wan error
Observation error
Observational error
Off-by-one bug

(continued)

Table 7.1 (continued)

Off-by-one error
Offset error
Omission error
Operative error
Overflow error
Override error
Parallax error
Parity error
Password error
Percentage error
Perception error
Permanent error
Persistent error
Personal error
Phenomenological error
Pilot error
Plain error
Polarization error
Position error
Positive spatial error
Prediction error
Printer error
Probability error
Probable error
Procedure error
Process error
Processing error
Program error
Programming error
Program-sensitive error
Projection error
Proportional error
Quantization error
Quantum error
Queue error
Random error
Ranging error
Read error
Reasoning error
Recoverable error
Rectification error
Refractive error

(continued)

Table 7.1 (continued)

Register error
Regular error
Rejection error
Relative error
Remote error
Renaming error
Representation error
Requirement error
Residual error
Residual standard error
Resolution error
Resource error
Retrieving error
Reversible error
Root-mean-square error
Rounding error
Round-off error
Runtime error
Sampling error
Scientific error
Semantic error
Sensitivity error
Sensor error
Sequence error
Sequencing error
Serious error
Server error
Set-up error
Simplification error
Simulation error
Single error
Soft error
Software design error
Software error
Solution error
Sorter error
Span error
Spatial error
Spatially correlated error
Specification error
Specification error
Specified error

(continued)

Table 7.1 (continued)

Spherical error
Spherical probable error
Stable error
Standard error
Standard error of estimate
Static error
Steady-state error
Subjective error
Substitution error
Syntax error
System error
Systematic error
Systemic error
System-level soft error
Table error
Target registration error
Technical error
Temporal error
Temporal prediction error
Temporary error
Tendency error
Tiny error
Tool error
Tracking error
Transaction error
Transcription error
Transfer error
Translation error
Transmission error
Transposition error
Trial-and-error
Truncation error
Tuner error
Type I error
Type II error
Type III error
Typical error
Typing error
Typographical error
Unbalanced error
Unbiased error
Uncorrelated error

(continued)

Table 7.1 (continued)

Undefined error
Unforced error
Unification error
Uninstaller error
Unintentional computing error
Unintentional error
Unknown error
Unknown hard error
Unrecoverable error
Unspecified error
Unstable error
Usage error
User error
User interface error
Variable error
Velocity error
Virtual device error
Virtual network device error
Willful error
Write error
Write protect error
Writing error
Zero error
Zero-scale error

Table 7.2 List of terms related with errors (terms are selected from the TBD informatics dictionary)

Automatic error correction
Automatic error detection
Barometric error code
Bit error rate
Circular error probability
Circular error probable
Err (v)
Erring
Error
Error analysis
Error bar
Error-based testing
Error checking
Error code
Error concealment

(continued)

Table 7.2 (continued)

Error control
Error-correcting
Error-correcting code
Error-correcting memory
Error correction
Error correction code
Error data
Error detecting
Error detecting code
Error detection
Error function
Error handler
Error handling
Error interrupt
Error log
Error message
Error propagation
Error rate
Error ratio
Error recovery
Error resilience
Error-bar chart
Error-based testing
Error-correcting
Error-correcting code
Error correction
Error correction qubit
Error free
Errorful
Erroring
Errorist
Errorless
Error-prone
Error-prone image Transmission
Error-tolerant
Error-tolerant system
Forward error correction
Functional error correction
Gauss error function
Law of error
Margin of error
More errorful

(continued)

Table 7.2 (continued)

Most errorful
Normal law of error
Optimal error rate
Plaintiff in error
Ultra-low-error rate
Ultra-low-error rate quantum system

Table 7.3 List of terms (other than error) related with failure (Terms are adapted from: Ören [3])

Accident
Amphibology
Blunder
Bug
Computer blunder
Counterfactual
defect
Delusion
Deviation
Disinformation
Erratum
Failure
Fallacy
Fallacy of composition
Fallacy of division
False acceptance
False alarm error
False alarm error
False document
False information
False negative
False news
False positive
Falsehood
Falsehood
Falsity
Fault
Faulty
Faux pas
Flaw
Formal fallacy
Glitch
Goof

(continued)

Table 7.3 (continued)

Howler
Inaccuracy
Inaccurate
Inadequate
Incorrect
Lapse of time
Lie
Malfunction
Malinformation
Material fallacy
Misapprehension
Miscalculation
Misconception
Miscue
Misdeed
Misinformation
Misinterpretation
Misjudgment
Mismanagement
Miss (v)
Misstep
Mistake
Misunderstanding
Malfunction
Noise
Noisy
Omission
Oversight
Paralogism
Shortcoming
Slight (v)
Slip-up
Slip-up (v)
Solecism
Sophism
Stumble (v)
Transgression
Unethical behavior
Untruth
Verbal fallacy
Weak
Weakness
Wrongdoing

7.2 Need for Reliability (Including Philosophical/Ethical Considerations About Reliability)

Valdemar Vicente Graciano Neto

Under the modeling and simulation framework (MSF) established for this BoK, *reliability* comprises the property of a simulation model (and its respective simulator) to be **reliable**, i.e., the *simuland* (system being simulated by a simulation, as seen in Sect. 1.4.1 of this BoK) should be enough detailed and accurate, and faithfully correspond to the system of interest being represented and/or developed. If the simulation model is reliable, conclusions and perceptions obtained from its execution can be reliably used to draw inferences about the simuland.

The need for reliability becomes even more evident during the engineering of large-scale and/or critical systems, i.e., those ones whose failures or malfunction could cause extensive financial losses, damage, and injuries to their users [6], such as aerospace, nuclear reactors, defense, crisis and emergency response, health applications, among others. Hence, simulations are demanded to anticipate systems properties, enabling corrections, and adjustments to make the resulting system equivalently reliable.

As seen in Sect. 1.4.4, models are *valid simplifications* that reduce the complexity to enable them to be executed on simulators. Briefly, providing reliability then involve confirming that each part being represented in the simulation model has a mapping counterpart in the simuland. Hence, reliability assurance inherently relies on conducting verification and validation (V&V) activities in regard to the simulation model. Rereading concepts presented in Table 1.6b, both V&V of a simulation model involve the simuland. From a philosophical perspective, a simulation model is validated and verified (and should be considered reliable, as a consequence) according to the degree of correctness it offers in regard to the system of interest being represented. Validation concerns to the degree to which a model with its associated data has an acceptable range of accuracy in regard to the system of interest in the real world as determined by the established experimental frame to support inferences on that simulation [7, 8]. In turn, verification consists of ensuring that the simulation model implementation is correct and is correctly executed [8]. Under MSF formalization, verification is a relation, called simulation correctness, between models and simulators; while validation is a relation, called validity in a frame, between models and real systems within an experimental frame. A simulation model should be validated and verified to be consistently reliable.

Techniques can be used to enhance the simulation model reliability. Particularly, V&V and inspection techniques can be adopted. Regarding simulation validation, an entire book has discussed in more than 1000 pages techniques for validation of computer simulation models [9]. A list of simulation model validation techniques is available in the literature (See a brief list in Sargent [8], and a broad, philosophical, and practical discussion on Beisbart and Saam [9]). Multi-resolution technique is also a technique to support validation by an examination of a simulation model under different granularities in order to assure its correctness against the real-world

counterpart system [10]. In turn, verification is primarily concerned with determining that the simulation functions (e.g., the time-flow mechanism, pseudo random number generator, and random variate generators) and the computerized (simulation) model have been programmed and implemented correctly [8]. Reference [11] mentions steps on the verification process, such as code verification and solution verification. A taxonomy of eight categories of techniques for simulation models verification is also available (See Whitner and Balci [12] for a quick summary of classic simulation model verification techniques).

7.3 Validation

Paul K. Davis

7.3.1 Introduction

The literature on model validation is massive, and this article is by no means a comprehensive review. It identifies unifying definitions, principles, and themes, but then points to important sources in varied research communities. The article is drawn from Davis [13].

7.3.1.1 Validity: Perhaps a Misnomer, but One We Can Live with

Models are imperfect and can never be given an unqualified stamp of validity. Further, working with models often goes far beyond just running a particular model that should be validated. As argued in by Jay Forrester in 1971 and reprinted later [14],

In any real-life applications of modeling to the generation of policy...the models are always in a continuous state of evolution. Each question, each reaction, each new input of information, and each difficulty in explaining the model lead to modification, clarification, and extension...

Rather than stressing the single-model concept, it appears that we should stress the process of modeling as a continuing companion to, and tool for, the improvement of judgment and human decision-making. [underline added]

Similar sentiments have been expressed in different domains of study, such as human-centered decision support in large organizations [15, 16]. One of the most valuable uses of simulation is in helping to educate senior leaders. Such leaders are often too savvy to imagine that the simulations are perfect, but they recognize the ability to learn from the simulations about the complex adaptive systems they deal with—enough to help them make better decisions currently and adaptively later as reality unfolds ([15, 17], p. 921).

All of this said, the term validation will not go away. Thus, we must live with it. The key is simply interpreting the validity of a model as its usefulness in some

particular effort. Does the model provides information that is good enough for the purposes at hand? The validation process should be understood as *testing* the model in many ways to see when (i.e., for what purpose and in what context) we can have an acceptable level of confidence in using the model. Excellent discussions on this exist in the literature.

A rigorous text discusses validation while recognizing the need to be “living with error” (Zeigler et al. [18], pp. 443–67). The text on system dynamics devotes 50 pages to testing of models (Sterman [19], pp. 843–92). Another text provides concepts and practical advice for military simulation (Tolk [20], pp. 263–94). Many other materials are readily accessible. Sargent and Balci [21] reviews the history of verification and validation (V&V), primarily for discrete-event simulation. See also Davis [22] Balci [23], Pace [24], Sargent [25] and a DoD best-practices guide and supporting material (Defense Modeling and Simulation Coordination Office (DMSCO), undated). The literature is meager on validation of agent-based models, but a Website includes sections on empirical validation <http://www2.econ.iastate.edu/tesfatsi/ace.htm>. The issues and challenges are discussed in short papers [26, 27], including the need to think in terms of incremental validation. Another paper emphasizes an approach that justifiably improves a client’s trust in the use being made of agent-based modeling [28].

7.3.1.2 Over Interpreting the Cautions

Many authors emphasize the conditional usefulness of models by repeating a famous quote from Box [29]:

All models are wrong; some are useful.

Regrettably, some of those hearing Box’s admonition take on a more negative attitude about models than is appropriate. It is not uncommon to hear something like the following from a modeler being pestered about whether his model has been tested.

Give me a break. No, I have not carefully validated the model. After all, it is only a model. It seems useful for what I am doing.

The phrase “only a model” should be troubling. If a model is being used to help understand the world and evaluate options for action, then it needs to be *good* for what it is being used for.

To reinforce this point, consider that some models are intended to be extremely close to reality. This is so for certain kinds of command and control systems where learning, training, mission rehearsal, and operations may use the same software, the same displays, and many of the same underlying calculations. Some exercise data may be synthetically generated (e.g., that of adversary aircraft or of approaching tornadoes), but the displayed information may be the same as if the aircraft or tornadoes were real. As another example, consider training of airline pilots. The best aircraft simulators are expected to exhibit aircraft behavior *very* similar to that of a real aircraft. This is why it came as a shock when, in 2019, Boeing acknowledged that its simulator for the Boeing 737-Max had not yet been able to

reproduce the problems that led to the crashes of two such aircraft [30]. In particular, pilots using the simulator for similar flight conditions would not have been prepared for the strength and persistence of the airplane's automated system forcing down the nose of the airplane when a sensor erroneously reported a troublesome angle of attack. As a last example, surgeons use virtual reality models to prepare for brain surgery [31]. Perhaps the models should be *very* good.

For most of those involved in modeling and simulation, it is nonetheless true that the models and their data are decidedly imperfect in some respects, and the issue becomes one of understanding whether they are sufficiently right in the respects that matter for the application.

7.3.1.3 Definitions

Against this background, there is need for corresponding formal definitions. These tend to address verification and validation as a set. The definitions used in this volume are (Zeigler, forthcoming). The U.S. Department of Defense uses similar definitions (Defense Modeling and Simulation Coordination Office (DMSCO), undated)

- *Verification* is the process of determining if an implemented model is consistent with its specification
- *Validation* is the process of determining if a model behaves with satisfactory accuracy consistent with the study objectives within its domain of applicability to the simuland it represents.

7.3.2 Distinctions

Primary Distinctions

Discussing validation is easier with a set of distinctions. A primary set of idealized distinctions involves the real world, the body of knowledge that we have about the real world, a conceptual model to describe implications of that knowledge, a fully specified version of that model suitable for implementation in a computer program, the mechanism by which the model's implications are generated (the program, computer program, computer model, or simulator), and the simulation results.

Figure 7.1 is a simple sketch of this that builds on an influential depiction by Sargent [32] and on points made by Zeigler [33, 34], Patrick Hester and Tolk [20], and Davis [35]. The diagram is to be read as follows: Given something to be represented—i.e., the *referent or simuland* (sometimes called the problem entity or real-world aspect), a *model* is developed—perhaps first in a conceptual form that captures the primary entities, relationships, and processes but is neither complete nor precise; and then in a fully specified but abstract form that provides a set of rules for generating model-predicted behavior for specified inputs. A *simulator* (computer program or computer model) then “executes” the model. The distinctions

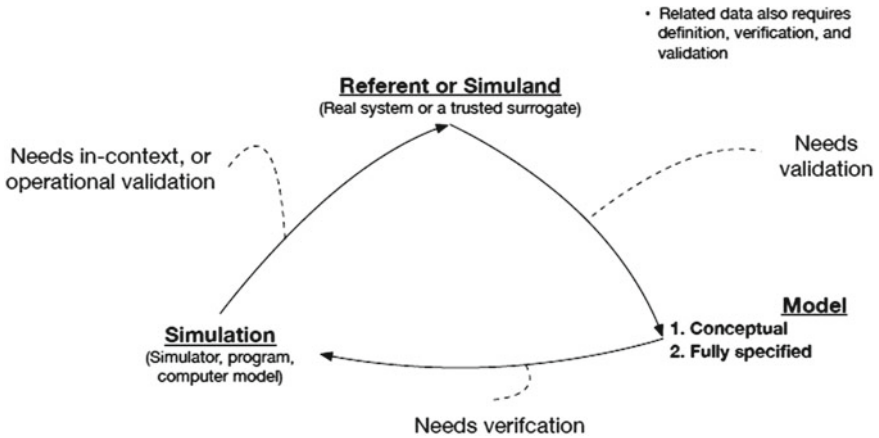


Fig. 7.1 Important distinctions

are valuable in understanding the many different ways that the activity of modeling can go badly: the knowledge base may be wrong, the model may represent the knowledge base poorly, or the program may fail to reflect the model's intentions. Unfortunately, authors use terms for the concepts of Figure 7.1 differently because work developed in parallel streams. Table 7.4 compares the terminology.

Defining Purpose: The Experimental Frame

If a model's validity (usefulness) depends on what it is being used for, then translating that broad idea into something more rigorous may require a good deal of detail. Precisely what variables will be held constant? Precisely what inputs will be provided? Precisely what outputs will be observed? And, for each, of these, what ranges of values will be considered? Recalling that "simulation" is often seen as experimentation (to others, the term refers to a model that generates system behavior over time), defining such matters can be seen as defining an *experimental frame* [18, 33].

A Fuller Depiction

Figure 7.2 shows a fuller depiction of the idealized distinctions and activities [35]. It is only modestly different from a diagram of Sargent [24].

This indicates myriad activities, in both the real and model worlds, to observe, experiment, and compare. When comparing model predictions with empirical information, it shows where the concept of the experimental frame fits in. It also indicates that inquiry may consider alternative conceptual models and alternative implementations. The former may be necessary because opinions differ on how the world works and how to represent it (e.g., with continuous differential equations, discrete-event simulation, agent-based modeling, or some combination). Also, competing simulations may differ in their simulators rather than their conceptual model.

Table 7.4 Differing terminology across sources

		Reference					
Term		Figure 7.1	Zeigler	Sargent and Balci	Sargent [25]	Tolk	Davis et al.
Referent or simuland		Referent	Simuland	Problem entity (system)	System problem	(Real system)	Real system
Surrogate for real world			(Simuland)	NA	System theories	System theories	(Real system)
Conceptual model		Conceptual model	NA	NA	Conceptual model	Conceptual model	Conceptual model
Fully specified model		Fully specified model	Model	Conceptual model	Simulation model specification	Formal model	Fully specified model
Simulation (simulator, program, computer model)		Simulation	Computerized model	Simulation model	Executable model; simulation system		Simulation model, simulation, program

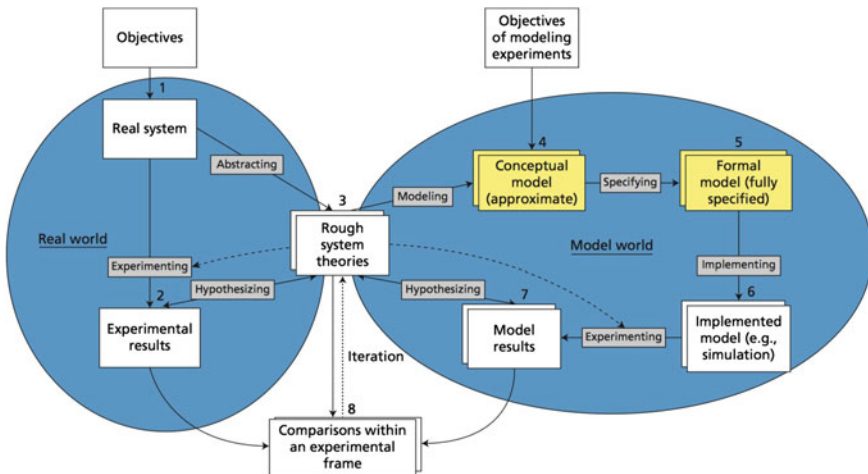
^aSee [18, 20, 21, 35]

^bZeigler et al. [18] also uses “simulator” to mean the mechanism for implementing any of a class of models, such as discrete-event, system-dynamic, or agent-based models

Results might, for example, differ because of differently sized time steps in continuous simulation or having agents act in a different order. A discrepancy might reflect a failure to adequately specify the model or a failure to implement the model correctly. Sometimes, differences have arisen because of unusual and embarrassing sensitivity of simulated events to computational details close to a boundary or discontinuity, as in instances where the same computer program has different answers when implemented on different machines [36]. Sometimes, such phenomena relate to the interaction of computer arithmetic and nonlinear effect, as in errors in the operation of the Patriot Air Defense System in 1991 [37].

Figure 7.2 is an idealization. In reality, it is rare for the conceptual model and the formal model to exist separately from the implemented model. One reason is that model builders use programming environments in which they can quickly “cut code” and build a running model with which to experiment. They may extoll the virtues of rapid prototyping in preference to prolonged agonizing in the abstract. However, a consequence is that it is difficult to understand and debate about either the conceptual model or a fully specified version. Instead, it is necessary to go immediately to often-opaque computer code or to trust documentation. This is a serious problem, but people disagree about how to do better.

One approach is to do initial modeling in a high-level visual language so that major features of the conceptual model can be readily reviewed visually and so that details and mathematics can be sharply expressed with the economy of array algebra. Such a model may then be used operationally or reprogrammed for a particular research environment [35]. A second approach, favored by most computer programmers, calls for better documentation, particularly of the conceptual



NOTE: Currently, the yellow items (conceptual and formal models) seldom exist separately.
RAND RR2208-5.2

Fig. 7.2 Idealized relationships. *Source* Davis et al. [35], but only modestly different from Sargent’s depiction [24]

model, while assuming that details will need to be worked out by people who are adept at computer programming in such common languages as

C, Java, R, and Python. A good protocol for documenting agent-based models has been proposed [38], and ambitious research continues in this spirit [39].

7.3.3 Generalizing the Concept of Validation

Types of Validity as Recognized Early

Historically, most discussion of model validation has reflected the perspective of physical scientists and engineers, such as those behind NASA's first flight to the moon. It was both necessary and feasible for the models guiding the moon landing to be remarkably accurate. In such a context, model validity is about the model's predictive accuracy.

Since at least the 1970s, however, some authors have distinguished among replicative, structural, and predictive validity [18, 33]. A model has replicative validity if it can mirror the referent system's input-output behavior, as when testing a model against empirical data. A simulation model has structural validity if its intermediate states and state transitions correspond to those of the referent system. A simulation model is said to have predictive validity if it can generate accurate predictions for circumstances other than those on which prior data exists.

The difference between replicative and predictive is referred to in social science work with terminology such as "the regression model fits the data reasonably well (a reasonably high R^2) versus "the regression model proved predictive when tested against out-of-sample data" (i.e., data not used in the model's formulation and calibration)." That distinction is crucial in modern artificial intelligence/machine learning.

The issue of structural validity is more knotty. It demands not just that the model gives the right answers, but that its internal processing relates well to real-world processes. That may or may not be important. For example, if a simulation correctly predicts equilibrium conditions, perhaps it is not important whether the simulation's intermediate states and processes are like those of the real world. Such was the argument of economist Milton Friedman, who argued that the economy behaved "as if" actors made decisions according to a simple model without much internal structure [40]. The arguments were strongly but politely criticized by another economist, Robert Samuelson [41]. In any case, for many purposes a simulation must have a significant degree of structural validity.

Broadening Scope of Information Used

One early generalization of the validation concept emphasized the need to consider a broad range of information in evaluating models. In the military domain, this include drawing on expert testimonies from officers experienced in combat, as well as on history, laboratory and field tests, and so on [22]. Figure 7.3 illustrates the range of information that can be brought to bear, but frequently is not. For example, model validation often relies heavily on face validation, i.e., on experts observing

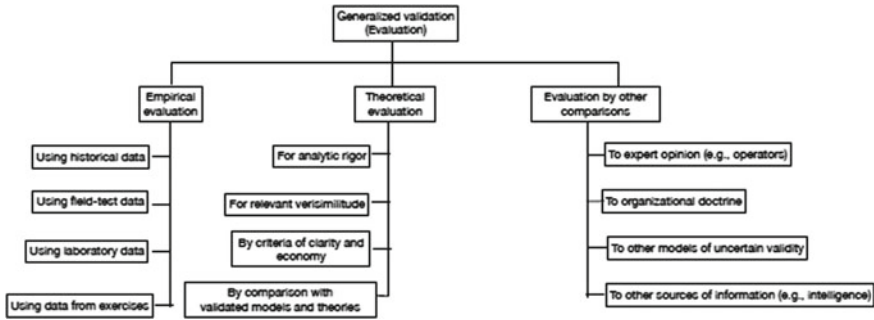


Fig. 7.3 Illustrative classes of information to use in validation

simulation behaviors and seeing them as credible, even though experts are far better at identifying the factors at work than at predicting consequences [42], much less the consequences of nonlinear interactions. The best use of subject matter experts is well structured and focused on eliciting the kinds of information that experts are good at providing [43].

Diverse Testing Methods

A more comprehensive approach recognizing the need for multifaceted evaluation of models can be found in the realm of system dynamics. From the pioneering work of Jay Forrester onward, those building such models have focused less on validation in its narrow sense conjuring up an image of precise calibration than to informing and testing the model against a broad range of information with a broad range of methods [44]. The authors noted that models are often used to look into matters for which no reliable data exists (e.g., the operation of future systems, conflict in future wars, side effects of innovative societal interventions). As a result, relatively, few of the tests can exploit the statistical methods that have been well developed for data-intensive testing.

Challenges for Validation as Seen in 2004

Early in the century, a good review article identified seven major challenges for model validation [24], all of which are still relevant despite notable progress. Pace's challenges were (paraphrased slightly):

1. Improving qualitative assessment (e.g., greater structure and rigor)
2. Use of formal assessment processes
3. Estimating costs and benefits of both M&S and V&V
4. Confronting and quantifying uncertainty and making inferences about domains for which no data exists
5. Coping with adaptation (e.g., validating agent-based models)
6. Aggregation and multi-resolution modeling
7. Human behavior representation in simulations (including by humans) (HBR).

7.3.3.1 Rethinking the Concept of Validation

It has been recognized for some time that the very concept of validation needs to be broadened if it is to be applicable in such realms as policy analysis and social science more generally. A recent review concluded that the definition of model validation can be improved in important ways by elaborating on the classic definition [35], pp. 23–31, 129–134). Adapting discussion to use this volume’s baseline definition, the improvement consists of adding the bolded language in what follows:

Validation: It is the process of determining if a model behaves with satisfactory accuracy consistent with the study objectives within its domain of applicability to the simuland it represents. ***A model’s validity should be assessed separately with respect to five criteria, the model’s capability for (1) description, (2) causal explanation, (3) postdiction, (4) exploratory analysis, and (5) prediction.***

That is, we should ask how well the model accomplishes the following:

- *Description*: identifying salient structure, variables, and processes in the target system (the system being modeled)
- *Causal explanation*: identifying causal variables and processes and describes reasons for behavior in corresponding terms;
- *Postdiction*: explaining past system behavior quantitatively with the benefit of knowing, afterward, the initial conditions pertaining at the time;
- *Exploratory analysis*: estimating approximate system behavior as a function of simultaneous changes in all identified parameters and variations of model structure (coarse prediction);
- *Prediction*: predicting system behavior accurately and even precisely;
- Prediction (the last item) is well understood, but the other items bear discussion.

Causal explanation is a familiar and primitive concept but is also deep and subtle [45, 46]. Postdiction (sometimes called retrodiction) may sometimes be after-the-fact rationalization but is very important in physics and say, in the diagnosis of why an aircraft crash occurred. Exploratory analysis refers to studying the model’s behavior across the space of uncertain inputs (also called scenario space or case space), the space generated by discretizing the input parameters and listing alternative model structures, and then considering the possibilities implied by the combinations of all possible values of all the parameters and choice of model structures. With today’s methods and technology, this is not as daunting as it may seem if the model used for exploration is relatively small (e.g., 3–20 independent variables, rather than 100s). Capabilities-based planning assesses alternative portfolios of capabilities (material, human, and otherwise) for their ability to address challenges across as wide a portion of the case space as is feasible within the budget [47]. Its cousin, robust decision-making (RDM) does similarly: searching for strategies that are as robust to assumptions as possible [48]. Such analysis can be used for a kind of coarse prediction, such as that a given strategy will do well (not necessarily optimally) in a range of circumstances and poorly (or disastrously) in a range of other circumstances.

These dimensions refer to different model *functions*. They relate to the previously mentioned distinctions among replicative, structural, and predictive validity, but not always in a straightforward way. Further, the relationships are confused by differences of terminology across disciplines. To those using statistical models, “explanation” may mean large coefficients of determination (R^2), whereas others have in mind cause-effect relationships. Similarly, whereas a statistician refers to predictive capability, the goal may be only to predict existing out-of-sample data (i.e., data not used to build and calibrate the model). Others have in mind predictions based on cause-effect reasoning that may be applied to circumstances very different from those on which data exists (e.g., descending through the Martian atmosphere, operating a future weapon system in a future war, or making a social intervention that will change the incentives of numerous actors in society).

The points causality being made here are closely related to those of Judea Pearl. As Pearl has notably observed, correlational information is insufficient to address many of the most important questions that face decision-makers, questions relating to influence, intervention, attribution, and so on. Pearl’s primary book on such matters is technical [49], but a more recent book is written for a broader audience. It has admirable examples [50]. He discusses what he calls a ladder of causation. The highest ladder corresponds roughly to the level of requiring a degree of structural validity (see also Zeigler [51]).

Figure 7.4 illustrates how a particular model might be characterized for its validity along these dimensions. The values {1, 2, 3, 4, 5} correspond to the qualitative meanings {very poor, poor, marginal, good, very good}. How these would be measured would depend on the application context. Prediction to within 10% might be very good in some cases and woefully inadequate in others. The

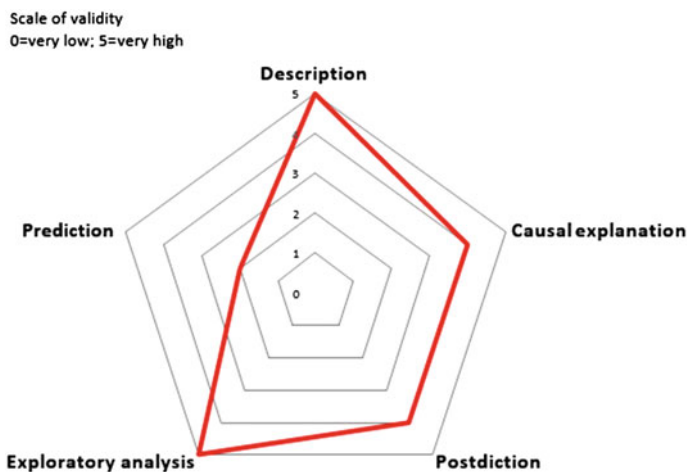


Fig. 7.4 Characterizing an illustrative model’s validity along five dimensions. *Source* Davis et al. [35]

assessment might need to combine qualitative and quantitative considerations (e.g., a causal explanation might seem subjectively to be too complicated to be useful, meriting a 1). The particular example shown imagines a model that is very good at describing what happened in a studied case, very good for broad exploratory analysis, rather good for causal exploration and postdiction, but of very little use for prediction. Why? Perhaps the input variables have highly uncertain values although the values might be well defined when the time comes. In contrast, certain artificial intelligence models based on machine learning can be extremely good for what those in that field call prediction (predicting data not used in tuning the model or future data of the same system under different conditions), but useless for assessing the consequences of options that would substantially change the nature of the system (e.g., options greatly changing the incentives to which actors in the system respond).

The concept of Figure 7.4 enhances our vocabulary for discussing the validity of models. Better methods are needed to assess models along each of these dimensions.

7.4 Verification

Bernard P. Zeigler

7.4.1 Introduction

Adhering to the definitions of Table 1.6a for verification and validation (V&V), this article focuses in on the problem of verification, particularly we discuss integration of simulation and formal verification as well the role of morphisms in providing the preservation of properties common to real systems and their simulation models.

We start with noting that concepts for organizing models and data for simulation based on systems theory [52, 53] and implementable in model-based systems engineering [54] are included in the modeling and simulation framework (MSF) (see Sect. 1.4). The system specification hierarchy (Sect. 1.4.1.1) provides an orderly way of establishing relationships between system descriptions as well as presenting and working with such relationships. Pairs of system can be related by morphism relations at each level of the hierarchy. A *morphism* is a relation that places elements of system descriptions into correspondence. For example, at the lowest level, two observation frames are isomorphic if their inputs, outputs, and time bases, respectively, are identical. In general, the concept of morphism tries to capture similarity between pairs of systems at the same level of specification. Such similarity concepts have to be consistent between levels. When we associate lower level specifications with their respective upper level ones, a morphism holding at the upper level must imply the existence of one at the lower level. The morphisms are set up to satisfy these constraints.

The most fundamental morphism, called homomorphism, resides at the state transition level consider two systems specified at level 3, S and S' , where S may be bigger than S' in the sense of having more states. S could represent a complex model and S' a simplification of it. Or S could represent a simulator and S' a model it is executing. If such a homomorphism holds for all states of S' , then any state trajectory in the S' will be properly reproduced in S . Often, we require that the correspondence holds in a step-by-step fashion and that the outputs produced from corresponding states be the same. In this type of homomorphism, the values and timing of the transitions and outputs of the big system are preserved in the small one. Thus, in this case, the state and output trajectories of the two models, when started in corresponding states, are the same.

Within the MSF, the EF formally recognizes that the intended use (IU) of a model is a fundamental determinant of its validity with respect to the source system. The MSF helps clarify many of the issues involved in modular reuse, validity, and executability of simulation compositions and V&V. The MSF underlies the DEVS simulation protocol [55, 56], which provides provably correct simulation execution of DEVS models, thereby obviating commonly encountered sources of errors in legacy simulations.

7.4.2 Intended Uses and Experimental Frames

Figure 7.5 shows the application of the framework to V&V. First, we consider that an organization with a heavy reliance on M&S for SoS, the US Missile Defense Agency is establishing a **standard IU specification** that contains a comprehensive

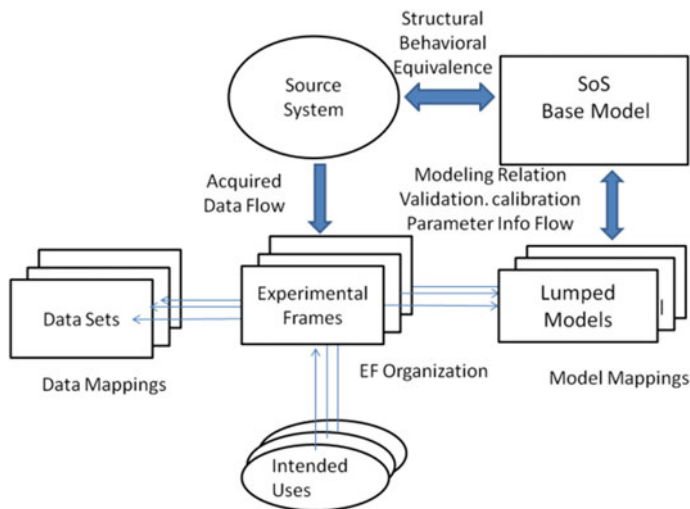


Fig. 7.5 Architecture for SoS V&V based on M&S framework

listing of specifics in relation to the analysis problem that the model is intended to address [57]. Indeed, this specification significantly expands the set of elements that characterize the objectives of the user. The fundamental elements of an IU include pertinent analyst tasks, model inputs and outputs, experimental designs, calibration methods and data, test objectives, and concepts of operations (CONOPS). In addition, the specification requires characterization of key attributes, i.e., aspects and values that identified stakeholders and developers agree on such as focus (from narrow consideration of a component (e.g., specific radar) to broad scope of the end-to-end SoS), simulation type (constructive, virtual, live), fidelity, uncertainty quantification, interoperability, level of detail, and relation to operator training or exercise experience.

Our formulation, based on such a specification, is that once the IU is known, suitable experimental frames can be developed to accommodate it. Such frames translate the IU elements into more precise experimentation conditions for the SoS base model and its various abstractions and aggregated models called *lumped models*. A model developed for an application is expected to be valid in each frame associated with the IU specification that formalizes that application. An IU specifies a focus, fidelity, and a level of detail to support the problems and tasks it concerns. Different foci, fidelities, and levels of details may both require and allow different models that exploit these factors to enable optimal set up and runtime attributes. ***The basic concept in Fig. 7.5 is that IUs act as keys to all data and models that have been acquired and developed thus far.*** In the storage process, a new data set or model is linked to the IU that motivated its development. In retrieval, given an application of interest to the user, the system supports formulating a representative IU and finding the closest IU matching the newly formulated IU. If the user is unsatisfied with the match, or wishes to explore further, the system supports synthesizing a composite IU using available lattice-like operations (upper and lower bounds, decomposition, etc.). [58] provide a metric-based method which aims to guide experimental frame and/or model definition to assist finding the right model and the right experimental frame for a given intended use.

7.4.3 Integration of Simulation and Formal Verification

Zeigler and Nutaro [59] discuss the use of morphisms that builds upon recent extensive work on verification combining DEVS and model checking for hybrid systems. The mathematical concepts within the DEVS formalism encompass a broad class of systems that includes multiagent discrete-event components combined with continuous components such as timed automata, hybrid automata, and systems described by constrained differential equations. System morphisms can map a model expressed in a formalism suitable for analysis (e.g., timed automata or hybrid automata) into the DEVS formalism for the purpose of simulation. Conversely, it is also possible to go from DEVS to a formalism suitable for analysis for the purposes of model checking, symbolic extraction of test cases, reachability, among other analysis tasks.

Humphrey [60] explored the use of linear temporal logic, the SPIN model checker, and the modeling language PROMELA [61] for high-level design and verification in unmanned autonomous vehicles (UAV)-related applications. She reported some success while suggesting limitations and needed extensions. Table 7.5 shows three UAV-related cases she discussed.

In each case, the focus of model is shown along with a simplifying assumption. Because they are oriented to verification, model checking tools tend to lack many functions that exist in DEVS environments and require abstractions that fit the tools' operation. This forces an abstraction of the real system that, on the one hand, enables the modeler to better understand the model, and on the other hand entails numerous assumptions to enable the model checker to verify the focal requirement. Despite these drastic simplifications, state space explosion prevents employing more than a handful of UAVs and sensors.

Several DEVS methodologies have been developed which incorporate non-DEVS verification methods [54, 62–64]. These methodologies attempt to employ DEVS to enable loosening the simplifying assumptions typically made by non-simulation models. In another variation, functional and temporal properties of a Timed Stream Petri Net models are checked using exhaustive verification or DEVS-based simulation [65]

The combination of simulation and formal verification gives a much more powerful capability to test designs than can be achieved with either alone. In a design process that incorporates both types of analysis, verification models can be used to obtain absolute answers concerning system behavior under idealized

Table 7.5 Example applications of model checking to a UAV multiagent system of system

Model: A centralized UAV controller that coordinates the actions of multiple UAVs performing a monitoring task
Focus of Model Checking: Assuring that all sensors are eventually visited
Sample Simplifying Assumptions: Communication between UAVs and sensors can only occur when in the same location and is error free
Model: A leader election protocol for a decentralized system of unattended ground sensors sending estimates of an intruder's position to a UAV
Focus of Model Checking: At least one leader exists at every time step
Sample Simplifying Assumptions: The sensors all use sampling epochs of the same length enabling a single time step for time advance
Model: Verification of high-level UAV mission plans for a scenario in which multiple UAVs must be used to safely escort an asset across a road network
Focus of Model Checking: The path traveled by the asset is safe, i.e., all road segments in the path have been scanned by UAV

conditions. Failures in this verification stage should indicate a need to find and correct fundamental flaws in the system design. *On the other hand, if a successfully verified model can be formally extended into a simulation model for which the verification model is a homomorphic simplification, the simulation model might retain the properties that were verified with the simpler model and then can be used to explore scenarios that are necessarily outside the scope of formal verification.*

The representation of this idea within the organization of relations is shown in Fig. 7.6. Here, the lumped model represents an analysis model that we are seeking to verify for a set of requirements and assumptions represented by EF_{Assum} . Further, the base model represents a simulation model that has more of the structure and behavior representative of the real-world system and accommodates a “larger” frame, EF_{Scope} . The fact that the latter is “larger,” i.e., more inclusive, than the latter is captured by the derivability relation. Then, if we can demonstrate a homomorphism from the base to the lumped model EF_{Assum} , we expect that any property proved to hold for the lumped model would also hold for the base model in that frame. However, it is more questionable whether the property continues to hold in EF_{Scope} .

To illustrate, let EF_{Assum} contains the first simplifying assumption of Table 7.5 “communication between UAVs and sensors can only occur when in the same location and is error free” and suppose the lumped model of the UAV system has been shown to meet the requirements that “all sensors are eventually visited.” Now suppose that in addition, we construct a base model that allows for realistic communication (less spatially constrained and error-prone). Under what circumstances would it be possible to show that all sensors are still eventually visited? Presumably, the base model would have to be extended in such a way as to have the “same” structure and behavior as the lumped model—which would require that a strong morphism hold between them. It would be natural to first consider that the morphism holds when the simplifying assumption is made (i.e., within EF_{Assum}), and then, whether it still holds when the condition is relaxed (i.e., within EF_{Scope}).

A full framing of the problem for SoS is obtained by returning to architecture of Fig. 7.5 where for the UAV multiagent SoS in Table 7.5, we have at least 3 EFs

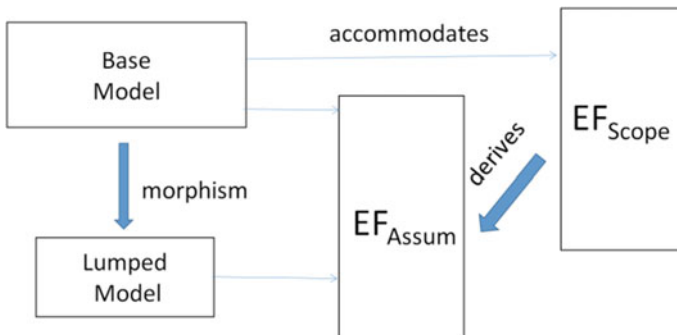


Fig. 7.6 Relation between verification and simulation

(each representing different simplifying conditions) and 3 lumped models (each representing a verified analyzable model). What can we say about a base (simulation) model that attempts to be compatible with each of the simplified models and therefore retain the desired properties they satisfy? The fundamental ordering relations and the system theory hierarchy of specifications and morphisms give us, at minimum, a means to frame the problem and develop a methodology to approach its solution.

In the last section, we address the question of what kind of morphic relation is needed between base and lumped models in order for proven properties of the lumped model to also hold for the base model.

7.4.4 Morphisms and Preservation of Properties

Consider base and lumped models with a homomorphism holding in an EF as in Fig. 7.6. Assume the lumped model has property P . Is it true that the based model must have property P as well?

Consider Fig. 7.6, where for convenience, we will use S (or system) to refer to the base model and M (or model) to refer to the lumped model. We call *upward* preservation or structural inference. The inference: “If M has P , then S has P ,” and as emphasized, it represents the kind of preservation we are focusing on here. However, *downward* preservation where a property P is inherited from S to M is of interest as well. The problem of downward preservation was raised by Foo [66, 67] who provided sufficient conditions for inheritance of stability properties for continuous systems and pointed out that downward preservation of P implies upward preservation of $\neg P$, the negation of P . Unfortunately, properties of interest seem not to be expressible in negative form. Indeed [68] clarified the situation by applying a well-known universal algebraic formulation of the logician Lyndon to finite automata. Informally, a positive property is one expressible in first order logic without use of negation. Conversely, a negative property is one that requires negation to express it. Sierocki enumerates a number of properties of automata that are of interest (e.g., relating to reachability, connectedness, and reversibility) and are positive by direct statement. Applying Lyndon’s theorem, Sierocki shows that upward inheritance of positive properties holds for the usual homomorphism of automata. Moreover, downward inheritance holds for negative properties.

The general situation is unsettled. Saadawi and Wainer [63] show that some properties transfer upwards from safety timed automat models verified in UPPAAL to real-time advance DEVS models under a strong form of bi-simulation similar to isomorphism. Zeigler et al. [69] provide examples of morphisms and properties where it is both possible and not possible to make structural inferences.

Given this situation, one direction for research is to look at more special cases. Another is to formulate the problem within a probabilistic, rather than logical, framework. It turns out we can get a more robust approach to making structural inferences as well as gaining more insight into the role of morphisms in the process.

7.4.4.1 Probabilistic Perspective: Bayesian Reasoning

The thin arrows in Fig. 7.7 denote pairing of systems and models corresponding to morphisms. Consider that in the situation for downward preservation, with the blue arrows, that systems having P map to models having P. The problem is caused by the possibility, shown as a gray arrow, that systems not having P also map to models having P. In other words, a morphism may be *deceptive* even though it preserves behavior and/or structure at some level of the hierarchy. Certainly, if we set up a random pairing of Ss and Ms where Ss with P always map to Ms with P, we would not expect that Ss without P also map to M's without P, However, there may be constraints on the Ss, Ms, and mappings that pull it away from pure randomness and make it less likely that a deceptive cross-mapping exists. Indeed, we can show that under conditions consistent with downward preservation of hard-to-prove properties, the posterior probability that S has P given that M has P is greater than the prior probability that S has P. In other words, demonstrating a homomorphism, or more generally a strong morphism, can significantly increase the likelihood that a lumped model property is truly reflective of the more complex base model [70].

7.4.5 Summary

The extended MSF provides a framework for V&V of simulation models that include the concept of intended use, a characterization of modeler objectives that enriches experimental frames. The MSF framework is applied to the case of integrating model checking with simulation where there is a need to have confidence that the properties proved for idealized models also hold in more realistic simulation models strongly related to them. Taking both logical and probabilistic perspectives clarify the situation and suggests that more research to introduce Bayesian reasoning to increase the robustness of V&V of simulation models.

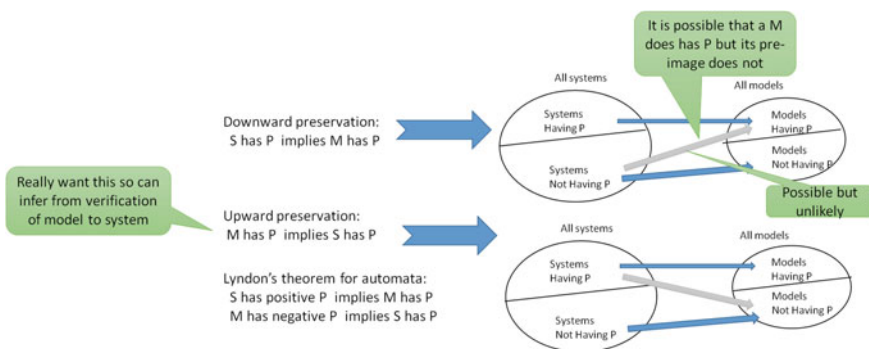


Fig. 7.7 Downward and upward preservation of properties

7.5 Failure Avoidance

Tuncer Ören

“Simulation, agent-directed simulation, and its synergy with systems engineering similar to many other advanced and sophisticated artifacts are prone to several categories of failures. Hence, to get full benefits of simulation, without undesirable error-prone side effects, one needs to consider sources of failures in M&S and have them under control. For this reason, V&V and more comprehensibly QA studies are done in M&S. However, failure avoidance for agent-directed simulation as well as for systems engineering should also be considered, especially in advanced M&S systems benefiting from the synergy of agent-directed-simulation and systems engineering” [71].

As failures of systems based on simulation studies are not desirable, validation (Sect. 7.2), and verification (Sect. 7.3) techniques aim to eliminate them. Some early publications promoted avoidance of failures of systems based on simulation studies. A classical article by Annino and Russell [72] was on “the ten most frequent causes of simulation analysis failure—and how to avoid them!” Even an earlier publication was about “don’ts of mathematical modeling” [73]. Sturrock’s contribution is titled: “avoid failures! tested success tips for simulation excellence” [74]. A systematization of the acceptability of components of simulation studies was given by Ören [75] and later generalized by Sheng et al. [76]. A further elaboration about the criteria for assessment as well as elements of M&S to be assessed is given by Ören and Yilmaz [71].

To assure reliability, it is useful to analyze sources of failures. Some basic concepts which might compromise reliability were outlined by Ören and Yilmaz [71]. They are failure, mistake, error, fault, defect, deficiency, flaw, shortcoming, sophism, and paralogism. Section 7.1 elaborate on sources of failures and lists 360 types of errors and many terms related with errors and failures.

Contribution of simulation to failure avoidance as well as need for multi-paradigm approach for successful M&S projects (including failure avoidance paradigm for successful M&S projects) is elaborated by Ören and Yilmaz [71]. Furthermore, failure avoidance is utmost important for the following types of M&S studies [71]:

- (1) agent-based modeling,
- (2) agent-directed simulation,
- (3) simulation where rule-based AI systems are used,
- (4) agents with personality, emotions, and cultural background
- (5) inputs (both externally generated inputs and internally generated inputs), and
- (6) systems engineering of M&S studies

In a follow up publication, Longo and Ören [77] elaborate on enhancing quality of supply chain nodes simulation studies by failure avoidance.

“For example, the supply chain nodes operation study can be enhanced in the following dimensions:

- *Role of terrorist activities*: Impact of two types of activities can be studied; to eliminate them or to alleviate their impact: (1) Using containers to smuggle material to be later used in terrorist activities within a country. (2) Impact of terrorist activities on the equipment of a supply chain node.
- *Global supply chain risk management simulations*.
- *Container scanning risk management simulation*.
- *Role of maintenance of several types of equipment*: Similar to the simulation studies of a job shop, several types of equipment in a supply chain node would require maintenance. The existing study can be extended for this purpose. Otherwise, the existing study may not be sufficient to analyze the need and allocation of resources for maintenance purposes.
- *Trend analyzes of the usage of the capacity of supply chain node*: Under different past conditions, the capacity utilizations and associated usage trends can be established. This information can be used in marketing the unused capacity; or coupled with simulation studies with anticipated demands can be used to perform investment analyzes” [77].

References

1. Carrie A (2021) The 8 most common root causes of medical errors. Always culture—HCAHPS (Hospital Consumer Assessment of Healthcare Providers and Systems) <https://alwaysculture.com/hcahps/communication-medications/8-most-common-causes-of-medical-errors/>
2. Mittal S, Durak U, Ören T (eds) (2017) Guide to simulation-based disciplines: advancing our computational future. Springer
3. Ören T (2020) Ethical and other highly desirable requirements for cyber-physical systems engineering. Chapter 17 In: Risco-Martin J-L, Mittal S, Ören T (eds) Simulation for cyber-physical systems engineering: a cloud-based context. Springer, pp 429–445 (in press)
4. Stanovich K (2009) Rational and irrational thought: the thinking that IQ tests miss. *Sci Am Mind* 20(6):34–39. <https://www.scientificamerican.com/article/rational-and-irrational-thought/>
5. AppSS (2020) Application Security Series, ImmuniWeb—AI for Application Security. <https://www.immuniweb.com/blog/top-10-failures-of-ai.html>
6. Manzano W, Graciano Neto VV, Nakagawa EY (2019) Dynamic-SoS: an approach for the simulation of systems-of-systems dynamic architectures. *Comput J*. <https://doi.org/10.1093/comjnl/bxz028>
7. Roache PJ (2009) Perspective: validation—what does it mean? *J Fluids Eng* 131(3):034503–034503-4. <https://doi.org/10.1115/1.3077134>
8. Sargent RG (2013) Verification and validation of simulation models. *J Simul* 7(1). <https://doi.org/10.1057/jos.2012.20>
9. Beisbart C, Saam NJ (2019) Computer simulation validation—fundamental concepts, methodological frameworks, and philosophical perspectives. Springer
10. Zeigler BP (2017) Constructing and evaluating multi-resolution model pairs: an attrition modeling example. *J Defense Model Simul* 14(4):427–437

11. Oberkampf WL (2019) Simulation accuracy, uncertainty, and predictive capability: a physical sciences perspective. In: Computer simulation validation. Springer, Cham, pp 69–97
12. Whitner RG, Balci O (1989) Guideline for selecting and using simulation model verification techniques. In: MacNair EA, Musselman KJ, Heidelberger P (eds) Proceedings of the 1989 winter simulation conference. IEEE, Piscataway, NJ, pp 559–568
13. Davis PK (draft) Model validation. In Davis PK (ed) Lectures in Policy Analysis and Modeling for complex systems. Pardee RAND Graduate School.
14. Forrester JW (1985) The model versus a modeling process (reprinted from 1971). *Syst Dyn Rev* 1(1)
15. Rouse WB (2019) Human-centered design of model-based decision support for policy and investment decisions. In: Davis PK, O'Mahony A, Pfautz J (eds) Social-behavioral modeling for complex systems. Wiley, Hoboken, NJ, US, pp 798–808
16. Rouse WB, Boff KR (2005) Organizational simulation. Wiley-Interscience
17. Davis PK (2019) Lessons on decision aiding for social-behavioral modelling. In Davis PK, O'Mahony A, Pfautz J (eds) Social-behavioral modeling for complex systems. Wiley, Hoboken, NJ
18. Zeigler BP, Muzy A, Kofman E (2018) Theory of modeling and simulation: discrete event and iterative system computational foundations, 3rd edn. Academic Press
19. Sterman JD (2000) Business dynamics: systems thinking and modeling for a complex world. McGraw-Hill, Boston
20. Tolk A (2012) Chapter 14: Verification and validation. In: Tolk A (ed) Engineering principles of combat modeling and distributed simulation. Wiley, Hoboken, NJ
21. Sargent RG, Balci O (2017) History of verification and validation of simulation models. In: Chan WKV et al (eds) Proceedings of 2017 winter simulation conference, pp 292–307
22. Davis PK (1992) Generalizing concepts and methods of verification and validation for military simulations, R-4249-ACQ. RAND Corporation, Santa Monica, CA
23. Balci O (1998) Verification, validation, and testing. In: Banks J (ed) The handbook of simulation. Wiley, New York, pp 335–393
24. Pace DK (2004) Modeling and simulation verification and validation challenges. *Johns Hopkins Appl Phys Lab Tech Dig* 25(2)
25. Sargent RG (2010) Verification and validation of simulation models. In: Proceedings of the 2010 winter simulation conference
26. Carley KC (2019) Social-behavioral simulation: key challenges. In: Davis PK, O'Mahony A, Pfautz J (eds) Social-behavioral modeling of complex systems. Wiley, Newark NJ
27. Carley KM (2017) Validating computational models. Institute for Software Research, Carnegie Mellon University, Pittsburgh
28. Hadzikadic M, Whitmeyer J (2019) Lessons from a project on agent-based modelling. In: Davis PK, O'Mahony A, Pfautz J (eds) Social-behavioral modeling for complex systems. Wiley, Hoboken, NJ, pp 655–671
29. Box GEP (1976) All models are wrong, but some are useful. *J Am Stat Assoc* 71(356):791–799
30. Kitroeff N (2019) Boeing 737 max simulators are in high demand. They are flawed. *New York Times* May 17. accessed at <https://www.nytimes.com/2019/05/17/business/boeing-737-max-simulators.html>
31. Roland J (2018) Virtual reality and medicine (next-generation medical technology), Reference point press
32. Sargent RG (1998) Verification and validation of simulation models. In: Medeiros DJ, Watson EG, Carson JS, Manivannan MS (eds) Proceedings of 1998 winter simulation conference, pp 121–130
33. Zeigler B (1984) Multifaceted modelling and discrete event simulation. Academic Press, Ontario, CA, 372 p
34. Zeigler B (2016) The role of approximate morphisms in multiresolution modeling: can we relax the strict lumpability requirements. *J Defense Model Simul*






35. Davis PK, O'Mahony A, Gulden T, Osoba O, Sieck K (2018) Priority challenges for social-behavioral research and its modelling. RAND, Santa Monica, CA
36. Dewar J, Gillogly J, Juncosa M (1991) Chaos and combat models, R-3995, RAND Corporation, Santa Monica, CA
37. Palmore JJ (1996) Research on the causes of dynamical instability in combat models, 96/95, US Army Corps of Engineers Construction Engineering Research Laboratories
38. Grimm V et al (2010) The ODD protocol: a review and first update. *Ecol Model* 221:2760–2768
39. Garibay I, Gunaratne C, Yousefi N, Schinert S (2019) The agent-based modeling canvas: a modeling Lingua Franca for computational social science. In: Davis PK, O'Mahony A, Pfautz J (eds) *Social-behavioral modeling for complex systems*, Wiley, Hoboken, NJ, pp 521–544
40. Friedman M (1966) *Essays in positive economics*. University of Chicago Press, Chicago, IL
41. Samuelson PA (1990) Problems of methodology discussion. *Am Econ Rev* 53:227–236
42. Tetlock PE (2017) *Expert political judgment: how good is it? How can we know?* New edn. Princeton University Press, Princeton, NJ
43. Balci O (2001) A methodology for certification of modeling and simulation applications. *ACM Trans Model Comput Simul* 11(4):352–377
44. Forrester JW, Senge PM (1996) Tests for building confidence in system dynamic models. In Richardson GP (ed) *Modelling for management simulation in support of systems thinking*, vol 2, Dartmouth Publishing, Aldershot, England, pp 414–434
45. Cartwright N (2004) Causation: one word, many things. *Philos Sci* 71(5)
46. Halpern JY (2016) *Actual causality*. The MIT Press, Cambridge, MA
47. Davis PK (2014) *Analysis to inform defense planning despite austerity*. RAND Corporation, Santa Monica, CA
48. Marchau VAWJ, Walker WE, Bloemen PJT, Popper SW (eds) (2019) *Decision making under deep uncertainty: from theory to practice*. Springer, Cham, Switzerland
49. Pearl J (2009) *Causality: models, reasoning, and inference*. Cambridge University Press, Cambridge, MA
50. Pearl J, Mackenzie D (2018) *The book of why: the new science of cause and effect*. Basic Books, New York
51. Zeigler BP (2019) How abstraction, formalization and implementation drive the next state in modeling and simulation. In: Sokolowski J, Durak U, Mustafee N, Tolk A (eds) *Summer of simulation: 50 years of seminal computer simulation research*. Springer, pp 25–39
52. Ören TI, Zeigler BP (2012) System theoretic foundations of modeling and simulation: a historic perspective and the legacy of A Wayne Wymore. *SIMULATION* 88(9):1033–1046
53. Wymore AW (1967) *A mathematical theory of systems engineering: the elements*. Wiley, New York, p 1967
54. Mittal S, Risco-Martín JL (2012) DEVS Net-centric system of systems engineering with DEVS unified process. *CRC-Taylor & Francis Series on System of Systems Engineering*
55. Nutaro JJ (2011) *Building software for simulation: theory and algorithms with applications in C++*. Wiley
56. Zeigler BP (1976) *Theory of modeling and simulation*, 1st edn. Academic Press
57. Piersall CH, Grange FE (2014) The necessity of intended use specification for successful modeling and simulation of a system-of-systems *CrossTalk*.2014
58. Foures D, Albert V, Nkesta A (2012) Formal compatibility of experimental frame concept and FD-DEVS model. In: 9th International conference of modeling, optimization and simulation —MOSIM'12
59. Zeigler BP, Nutaro JJ (2014) Combining DEVS and model-checking: using systems morphisms for integrating simulation and analysis in model engineering, *EMSS 2014 Bordeaux—France*

60. Humphrey LR (2013) Model checking for verification in UAV cooperative control applications recent advances in research on unmanned aerial vehicles. *Lect Notes Control Inform Sci* 444:69–117
61. Holzmann GJ (2004) *The SPIN model checker: primer and reference manual*. Addison Wesley Publishing Company
62. Denil J (2013) *Design, verification and deployment of software intensive systems: a multi-paradigm modelling approach*, Ph.D. dissertation, University of Antwerp
63. Saadawi H, Wainer G (2013) Principles of discrete event system specification model verification. *SIMULATION* 89:41–67
64. Zeigler BP, Sarjoughian HS (2012) *Guide to modeling and simulation of systems of systems*. Springer
65. Ciciirelli F, Furfaro A, Nigro L (2012) Using time stream Petri nets for workflow modelling analysis and enactment *SIMULATION* 2013 89: 68 originally published online 24 January 2012. <https://doi.org/10.1177/0037549711434603>
66. Foo N (1977) Stability preservation under homomorphisms. *IEEE Trans S.M.C.* 7(10):750–754
67. Foo N (1979) Homomorphisms in the theory of modeling. *Int J Gen Syst* 5(1):13–16
68. Sierocki I (1986) A note on structural inference in systems theory. *Int J Gen Syst* 13(1):17–22. <https://doi.org/10.1080/03081078608934951>
69. Zeigler, BP, Muzy A, Kofman E (2019) *Theory of modeling and simulation: discrete event & iterative system computational foundations*, 3rd edn. Elsevier
70. Zeigler BP, Nutaro JJ (2016) Towards a framework for more robust validation and verification of simulation models for systems of systems. *J Defense Model Simul* 13(1):3–16
71. Ören TI, Yilmaz L (2009) Failure avoidance in agent-directed simulation: beyond conventional V&V and QA. In Yilmaz L, Ören TI (eds) *Agent-directed simulation and systems engineering*. Systems Engineering Series, Wiley-Berlin, Germany, pp 189–217
72. Annino JS, Russell EC (1979) The ten most frequent causes of simulation analysis failure—And how to avoid them! *Simulation* 32(6):137–140. <https://journals.sagepub.com>; <https://doi.org/10.1177/003754977903200605>
73. Golomb SW (1970) Don'ts of mathematical modeling. *SIMULATION* 14(4):198
74. Sturrock DT (2017) Avoid failures! Tested success tips for simulation excellence. In: Chan WKV et al (eds) *Proceedings of the 2017 winter simulation conference*, pp 588–596. <https://www.informs-sim.org/wsc17papers/includes/files/043.pdf>
75. Ören TI (1981) Concepts and criteria to assess acceptability of simulation studies: a frame of reference. *CACM* 24(4):180–189
76. Sheng G, Elzas MS, Ören TI, Cronhjort BT (1993) Model validation: a systemic and systematic approach. *Reliab Eng Syst Saf* (Elsevier) 42:247–259
77. Longo F, Ören TI (2010) Enhancing quality of supply chain nodes simulation studies by failure avoidance. In: *Proceedings of the EMSS 2010–22th European modeling and simulation symposium (within the IMMM-7—The 7th International Mediterranean and Latin American modeling multi-conference)*, 13–15 Oct, Fes, Morocco, pp 201–208
78. Wikipedia: Hofstede's cultural dimensions theory. https://en.wikipedia.org/wiki/Hofstede%27s_cultural_dimensions_theory
79. Zeigler BP (forthcoming) “Chapter 1. Preliminary,” in *Modeling and simulation book of knowledge*. Society for Computer Simulation
80. TBD informatics dictionary: Dictionary of Turkish Informatics Society—TBD <http://bilisimde.ozenliturkce.org.tr/onerilen-tum-terimler-ingilizce-turkce/>



Ethics

8

Nico Formanek , Juan Manuel Durán , Paul K. Davis ,
Andreas Tolk , and Tuncer Ören 

Abstract

Ethics is a branch of philosophy that studies moral problems. This chapter of the SCS M&S Body of Knowledge starts with a broader look at how ethics are influencing technical disciplines in general before looking at ethics for computer simulation in more detail. It devotes a section on ethics for simulationists and analysts, as they often provide guidance for decision makers with significant consequences. The chapter concludes with the simulationists code of ethics.

Keywords

Modeling and simulation · Ethics · Code of ethics

N. Formanek (✉)
High Performance Computing Center, Stuttgart, Germany
e-mail: nico.formanek@hls.de

J. M. Durán
Delft University of Technology, Delft, Netherlands
e-mail: j.m.duran@tudelft.nl

P. K. Davis
The RAND Corporation and the Pardee RAND Graduate School,
Santa Monica, CA, USA

A. Tolk
The MITRE Corporation, Charlottesville, VA, USA
e-mail: atolk@mitre.org

T. Ören
University of Ottawa, Ottawa, ON, Canada

8.1 Branches of Ethics

Nico Formanek, Juan Manuel Durán

8.1.1 Ethics in Technical Disciplines

Ethics has grown into many different branches, treating increasingly specialized but also interdisciplinary problems. There are ethics of engineering, ethics of computers, and ethics of medicine—just to name a few branches. In general, we will call them ethics of X, where X can be basically any field where ethical problems emerge. To call them the ethics of X presupposes that each branch purports different and disjoint ethical issues. Thus, the ethics of engineering is concerned with issues alien to the ethics of computer simulations. But in practice, it is quite common to find shared concerns among all these branches of ethics. In this context, there are different connections between, say, the ethics of engineering and the ethics of computer simulation. If one thinks computer simulation as a sub-discipline of engineering, then the ethics of engineering will treat more general problems arising from engineering practice without reference to computers, while the ethics of computer simulation will be an application of the ethics of engineering to computer science.

A general problem which is treated in the ethics of engineering is the problem of unintended consequences. Every technology is designed with a specific end. To reach this end desired and undesired impacts are accounted for. It is obvious that not every effect of a technology, desired or undesired, can be predicted in the design process. For example, the use of fossil energy on a vast scale, which emerged during the industrial revolution, has as we now know some very undesirable side effects. Those were certainly not intended, nor known, by the early innovators constructing steam engines.

A similar problem will of course arise with every technology, one of which happens to be computer simulation.

Before we consider the special case of the ethics of computer simulation, let us talk briefly about what ethics and ethical problems are in general.

Ethics is the branch of philosophy that studies moral problems, that is, problems of right and wrong action. It is thus closely connected to the philosophy of action. For the purposes of this article, it is sufficient if we remain close to a common-sense concept of what an action is: One *does* something under self-determining conditions of possibility (e.g., one is not being forced to take some action). Actions and their consequences can be evaluated, and the task of an ethics of X is to give reasons for evaluating the special class of actions picked out by X. The scenario is like this: If you ask yourself “What is the best action in situation A, is it *F* or *G* or ...?” then the ethics of X will be a reservoir of reasons to pick out the best possible action—or so it is expected/desired.

For example, the ethics of medicine treats concerns about medical interventions on humans. Questions evaluating the action of the relevant stakeholders might be:

How does a physician weigh the needs of patients suffering from an illness in a randomly controlled study? Should the cheapest but less effective treatment be chosen, or rather the most expensive but also most effective treatment be chosen?

It should be noted that there is a more general classification of ethics in philosophy. Standardly, they are classified as consequentialism, deontology, virtue ethics, and pragmatic ethics. An ethics of X will employ one or more of these frameworks to evaluate the problem at hand. A perhaps too simplistic description of those frameworks would be that consequentialism only evaluates an action with respect to its consequences, deontology according to rules for carrying out those actions, virtue ethics corresponding to the virtue of the agent, and lastly pragmatic ethics evaluates actions according to the wider context in which they occur.

Consider the following example on how an ethical evaluation could work in the different frameworks.

Situation: You are in a hurry to get a job interview and are stuck in traffic. There is a shortcut, but you have to pass wrongly through a one-way road.

Question: Should you take the shortcut?

Actions: (A) Take the shortcut. (B) Don't take the shortcut.

Below are answers that certain ethical frameworks could give.

Consequentialism: Act always to maximize utility. A maximizes utility. Answer: Do A.

Deontology: Act always according to rule R. Action B satisfies rule R. Answer: Do B.

Virtue ethics: Act always to preserve your virtuousness. Action B preserves virtuousness. Answer: Do B.

Pragmatic ethics: The one-way road is rarely used by cars and taking it would reduce your stress levels. Answer: Do A.

Ethical frameworks do not generally provide unique answers about what to do. This has several reasons. Firstly, the frameworks themselves are justified by adducing artificial situations (sometimes more or less so—think of trolley problems), which makes picking out the adequate framework dependent on the situation description and the moral intuition underlying said description [1].

Secondly, even for non-pragmatic frameworks, the provided answers depend on the preselected actions. Those actions are generally not picked out according to some specified rule in the ethical framework, they rather depend on what the person doing the ethical evaluation is willing to admit. The kind of arguments that these frameworks supply for or against an action are at the very least *enthymematic*; in other words, in most cases it is not unclear if these frameworks can provide deductive certainty at all about which action to choose.

It is the uncertainty in the description of situations which connects ethics to other branches of philosophy like epistemology and philosophy of science. One of the biggest problems in ethics is how to make a morally sound decision under uncertainty.

You will notice that our later examples from the ethics of computer simulation could all be labeled as decision under uncertainty. While it would be nice if philosophy could reduce the uncertainty in the situation description, this is in many cases not possible. Uncertainty might even be the property of a situation

description, e.g., limited time and mental capacity of a person to adequately evaluate all presented options.

8.1.2 The Ethics of Computer Simulation

With this in mind, we now turn to ethics of computer simulation. First, some foundations have to be laid. We would not say what a computer simulation is, but rather what can be done with it. A computer simulation can run on a computer to compute and obtain *simulation results*. These results give an answer to a previously—however vaguely—stated question. The quality of the answer depends on many factors, internal and external to the simulation. One external factor is the specificity of the question according to which the computer simulation was built. More specific questions lead to better models, which in turn lead to better computer simulations. An internal factor would be the quality of the program. Does it contain many hacks, kludges, etc., which might affect its representational qualities?

In philosophy of science, models have for long been an object of inquiry. Models try to represent a part of the world and might include a number of idealizations, abstractions, and fictionalizations in order to do this. Uncertainty, then, is already introduced at these stages if it is unknown how those idealizations, abstractions, and fictionalizations affect the representational capacity of the model. The same is true for computer simulations with one *addendum*: computer simulations typically introduce more and different sources of uncertainty. This will become apparent in later examples. Among other things epistemology evaluates why some forms of uncertainty are tolerable while others might not.

Now, computer simulations would not be an interesting case for ethics if they did not figure prominently in ethical questions. And this is where the current literature on the topic takes its starting point. Everyone knows cases where simulation results have been used to justify policy decision. Examples include the IPCC report on global climate and the simulation of pedestrian traffic preceding the approval of the 2010 love parade in Germany.

So, whenever simulation results are used in situation descriptions for ethical questions this elevates the uncertainty of those results from a “mere” epistemological concern to an ethical issue.

Following [2] Chap. 7 and [3], I will now discuss several frameworks that have been proposed to cope with the uncertainty of simulation results in ethics.

According to Williamson [4], simulation results must be *trustworthy* if they are used in ethical decisions. Trustworthiness itself depends on several ethical and epistemic factors. For a simulation result to be trustworthy, it has to be *credible*, *transferable*, *dependable*, and *confirmable*. Williamson takes credibility to be established by inter-subjective methods of verification and validation, but also by expert authority. It is therefore a mixed epistemic and ethical concept to reduce the possible uncertainty inherent in the simulation results. The rest of the concepts are epistemic in nature. Transferability is the possibility of extending simulation results

beyond their original context. A situation that often happens in policy decisions that should apply across different situations.

Dependability is the property of simulation results to apply after a certain amount of time passed. Uncertainty might arise due to the target system changing in unknown and unaccounted for ways.

Williamson's last criterion of confirmability amounts to concerns about different idealizations that were introduced into the computer simulation, for example, idealizations that make the problem computationally tractable in the first place. Such idealizations can introduce uncertainties in the simulation results if they are not properly accounted for.

In the end, if uncertainties are present, they taint the ethical decision that rests on the simulation result, possibly leading to unethical choices of action.

A similar point is made by Brey [5], for whom uncertainty enters through misrepresentations. Computer simulations can represent or fail to represent a phenomenon, depending for example on which idealizations were in place during their implementation. Instances of misrepresentation might be hard to detect because direct comparison to experimental data is impossible. This epistemic concern again threatens ethical decision-making with uncertainty.

As we saw earlier, the authors of the quoted studies on computer simulation ethics are concerned with harm that might arise from ethical decisions which are based on uncertain simulation results. It is very hard to say what could be done to reduce the uncertainties, which is not bordering on platitudes like "improve verification and validation procedures." The most general kind of advice that is given in the existing literature is contained in codes of conduct.

Ören et al. [6] proposes such code of conduct specifically for computer simulations which is also described in Sect. 8.3 of this book.

In general, codes of conduct follow the spirit of virtue ethics or deontology. They provide rules for action or guidance on how virtuous conduct can be achieved. Ören's code is adapted to the needs of simulationists and thus applies only to ethical questions concerning the genesis, running, and use of computer simulations. The justification of the rules of the code depends on more general principles of good scientific conduct, best practices from programming, and previous codes of conduct for the engineering discipline.

8.2 Ethics for Simulationists and Analysts Using Modeling and Simulation

Paul K. Davis, Andreas Tolk

The rationale for addressing ethics in this volume on modeling and simulation has several components. For engineers, the basic rationale is that engineers *build things* that change the world. In doing so, they assume responsibilities to individual, organizational, and government clients, and to humanity in the large. Sometimes, the obligations are in conflict, which creates difficult tensions. Scientists, who often

use M&S, have obligations such as the search for truth and advance of science but also obligations to the people and even animals who participate in or are subjects of research. A third category of users consists of *analysts*. These may also be scientists or engineers, but they aid decision-makers and often support activities affecting people and the world. Those who build models have the obligation to make it possible for analysts to use the models well, correctly and wisely inform decision-makers, and assure fairness, and minimize harm. This article addresses, in turn: (1) definitions, (2) ethics in the modeling and analysis cycle, (3) why such ethics matter, (4) approaches to ethics, and (5) the role of professional codes.

8.2.1 Definitions

A recent textbook covers definitions, distinctions, and comparisons. It then has a number of concrete examples that illustrate vividly the ethical issues that arise for engineers [7]. Most of its material applies also to those associated with science, technology, and analysis. This article draws also on ideas in other published papers. For example, an early text laid much groundwork that is still very relevant [8] and the need for simulationists to have a code of conduct was discussed in an influential conference paper [9].

The definitions of morals and “ethics” are often used interchangeably. Distinctions are sometimes drawn, but in contradictory ways. Here, we use:

Ethics, also Called Moral Philosophy, is the discipline concerned with what is morally good and bad and morally right and wrong. The term is also applied to any system or theory of moral values or principles. (<https://www.britannica.com/topic/ethics-philosophy>)

In making distinctions, we use the formula that

Ethics are the science of morals, and morals are the practice of ethics. (Fowler and Crystal [10])

The adjectives “ethical” and “moral” can also be ambiguous, but “moral” usually refers to personal matters whereas “ethical” is favored when referring to matters of, e.g., medicine, law, science, or business.

8.2.2 Ethics in the Cycle of Modeling and Analysis

Why does ethics matter in a volume on modeling and simulation? Adapting a concept laid out in the text mentioned above [7], we note that ethical considerations are or should be important in each stage of the cycle shown in Fig. 8.1. The top line shows the process from problem definition to the delivery of well-articulated evaluation of options. Feedbacks (shown as dashed lines) occur throughout the process. For example, as options emerge, one recognizes the need to consider additional objectives with corresponding metrics. Also, when comparing options,

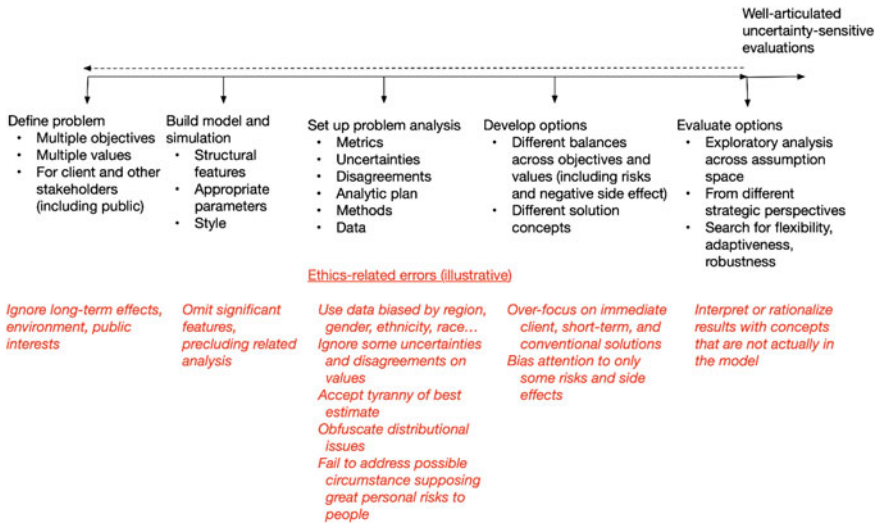


Fig. 8.1 Ethics in the cycle of modeling and analysis

one may be sensitized to uncertainties that should be explicitly addressed in the analytic plan.

As indicated by the italic material at the bottom, numerous ethical issues arise or should arise at every step. These are *illustrated* by the items shown, which indicate only some of the many ethical errors or lapses that may occur, such as ignoring long-term effects on the environment or the public's interests in privacy [11], doing the analysis with biased data [12], or obfuscating risks and distributional issues (as in dwelling only on average economic effects). Some other examples are (1) omitting key variables, which precludes correctly analyzing their effects (e.g., omitting the possibility of a tax cut's stimulus effect), and (2) explaining results based on concepts not actually in the model (e.g., ascribing an intention to a model object that merely follows some rules, oblivious of intention).

8.2.3 Why Ethical Considerations Matter

Most readers may find the importance of such matters evident, but a few examples may be worthwhile. Consider urban planning that focuses entirely on economic rejuvenation. The results may include destroying neighborhoods and cultural features, depriving people of their life-long homes, forcing such people to move to more hostile but affordable areas, and generating a “sterile” downtown without character. Such obliviousness to the many dimensions of the problem might be seen as incompetence, but not if the only consideration was stimulating economic growth in the downtown area. Or consider developing a simulator for a new aircraft, a simulator that is exceedingly accurate for most conditions but does not address

some plausible circumstances that would be expensive to understand and represent well. Pilot training in such simulators would not be prepared if the trouble circumstances arose. This occurred in the notorious case of failures of the Boeing 737-MAX. (A newspaper account touched high points [13], but more definitive accounts of the fiasco are slowly emerging [14]. The aircraft's failures killed 346 people. Many other examples could be given [7, 15].

One of the earliest discussions of ethics in the context of simulation was a paper by John McLeod, the founder of the Society for Modeling and Simulation [16]. McLeod was commenting on the danger that some use of simulation might be analogous to that of the accountant who “when asked ‘How Much is $2 + 2$?’ replied ‘How much do you want it to be?’” McCleod went on to provide draft ethical guidelines that emerged from a study by the National Science Foundation. Many other references might be named, each with own bibliographies (e.g., [7, 15]) A recent paper illustrates with critical review the important ethical subtleties that arise when attempting to address social issues with simulation [17].

8.2.4 Approaches to Applying Ethics

It is sometimes useful to distinguish among three different approaches that scholars take in addressing issues. The exact labels vary, but the three approaches are (1) consequentialist (utilitarian); (2) deontological (duty-driven as with adherence to laws, norms, or principles); and (3) virtue-seeking (seeking good character traits, such as reliability, honesty, ...). These are, roughly, associated, respectively, with Jerome Bentham and John Stuart Mill, Emanuel Kant, and Aristotle. They are discussed and compared, with examples, in Van de Poel and Royakkers [7].

8.2.5 The Role of Professional Codes

Many ways exist for addressing ethical considerations, but in this volume, we address only one, having professional organizations adopt codes of conduct.

Ethical codes can be crafted to be inspirational, advisory, or disciplinary in nature [18, 19]. Numerous examples exist, as well as a corresponding literature. Here, we merely touch upon examples.

An inspirational expression of engineering ideals is the oath taken to join the *Order of the Engineer*:

I am an Engineer; in my profession I take deep pride. To it I owe solemn obligations.

Since the Stone Age, human progress has been spurred by the engineering genius. Engineers have made usable Nature's vast resources of material and energy for Humanity's benefit. Engineers have vitalized and turned to

practical use the principles of science and the means of technology. Were it not for this heritage of accumulated experience, my efforts would be feeble.

As an Engineer, I pledge to practice integrity and fair dealing, tolerance and respect, and to uphold devotion to the standards and the dignity of my profession, conscious always that my skill carries with it the obligation to serve humanity by making the best use of Earth's precious wealth.

As an Engineer, I shall participate in none but honest enterprises. When needed, my skill and knowledge shall be given without reservation for the public good. In the performance of duty and in fidelity to my profession, I shall give the utmost.

(the Oath is copyrighted by the Order of the Engineer, Inc.)

To be sure, not all engineers take the oath, and not all that do necessarily live up to it in all respects, but the oath reflects an ideal with which many can resonate and to which many make every effort to adhere.

Advisory professional codes provide guidelines that help the simulationist to make good decisions, often very similar to Code of Best Practices. Many codes of professional conduct advise society members how to behave professionally. The IEEE Code of Ethics—documented in the IEEE Policies, Sect. 7: Professional Activities (Part A: IEEE Policies)—falls into this category.

We, the members of the IEEE, in recognition of the importance of our technologies in affecting the quality of life throughout the world, and in accepting a personal obligation to our profession, its members and the communities we serve, do hereby commit ourselves to the highest ethical and professional conduct and agree:

- 1. to accept responsibility in making decisions consistent with the safety, health, and welfare of the public, and to disclose promptly factors that might endanger the public or the environment;*
- 2. to avoid real or perceived conflicts of interest whenever possible, and to disclose them to affected parties when they do exist;*
- 3. to be honest and realistic in stating claims or estimates based on available data;*
- 4. to reject bribery in all its forms;*
- 5. to improve the understanding of technology; its appropriate application, and potential consequences;*
- 6. to maintain and improve our technical competence and to undertake technological tasks for others only if qualified by training or experience, or after full disclosure of pertinent limitations;*

7. *to seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, and to credit properly the contributions of others;*
8. *to treat fairly all persons regardless of such factors as race, religion, gender, disability, age, or national origin;*
9. *to avoid injuring others, their property, reputation, or employment by false or malicious action;*
10. *to assist colleagues and co-workers in their professional development and to support them in following this code of ethics.*

Disciplinary codes impose negative consequences for violations of standards. The consequences may include, e.g., paying fees for not disclosing conflicts of interest, exclusion from certain types of contract competition because of past violations, or being removed from the professional society.

Although many professional science and engineering societies have their own code of ethics (e.g., that of the Association for Computer Machinery [20]), certain elements are common to most of them, such as pursuit of truth, protection of community and environment, accountability for actions, mentoring the next generation, and informing and engaging the public. Recently, diversity and integration of minorities have been recognized as valuable goals for fairness because they allow new perspectives and ideas suggesting better solutions supporting society. Rigor, respect, responsibility, honesty, and integrity have been identified as the core values for scientists and engineers, including simulationists.

It has been noted that policy analysts do not have an ethical code and that it would be difficult to develop a sensible code. Douglas Amy stated “Ethical inquiry is shunned because it frequently threatens the professional and political interests of both analysts and policymakers. The administrator, the legislator, the bureaucracy, and the profession of policy analysis itself all resist the potential challenges of moral evaluation” [21]. Others have long argued otherwise and have suggested a code of conduct [22, 23]. A book on the subject [24] includes examples and recommendations.

8.2.6 A New Obligation for Those Who Build M&S and Use It for Analysts

It has long been an ethic that analysts identify the assumptions on which their results depend. Much more is necessary. Analysts should routinely discuss how results vary with major assumptions on which there is uncertainty or disagreement. This should reflect exploratory analysis in which assumptions are varied simultaneously, rather than mere variable-at-a-time sensitivity analysis. Further, analysts should demonstrate ways in which clients can hedge against uncertainties, i.e., how to identify strategies that are relatively more **Flexible** (to changes of mission),

Adaptive (to changes of circumstance), and **Robust** (to adverse shocks). This is sometimes referred to as planning for FARness [25, 26] or as what is becoming widely known as supporting Robust Decision-Making (RDM) under deep uncertainty [27, 28]. Such efforts should become an ethical obligation.

To put the matter differently, the analyst should go well beyond so-called best-estimate calculations (which are often misleading because of uncertainties) and indicate the range of circumstances under which the consequences of the strategies being considered are relatively predictable and favorable, relatively predictable and bad, or very uncertain and therefore risky [26]. M&S should be designed so as to make related analysis easier and routine. Failure to do such analysis may leave decision-makers with inappropriate confidence in best-estimate results, which may lead to seriously harmful decisions.

8.2.7 Final Observation

Today's simulations are powerful computational tools that can be seen as the third pillar of science [29], along with theory and empirical data. When used with visualization tools and augmented reality, they allow immersion into the problem space and direct interactions with the model. This vividness, however, can deceive a user that into seeing the simulations as valid surrogates of the real system when they are not. The ethical responsibilities of simulationists and those who use simulations are growing in parallel to these technological advances.

8.3 Code of Ethics for Simulationists

Tuncer Ören

The code of ethics for simulationists (as posted at <https://scs.org/ethics/>) has been developed by the following members of the Ethics committee of the SCS:

- Prof. Emeritus Tuncer I. Ören (Chair)—SCS AVP Ethics§ Founding Director of M&SNet—McLeod Modeling & Simulation Network of SCS
- Prof. Emeritus Maurice S. Elzas, Wageningen Univ., Wageningen, The Netherlands
- Prof. Emeritus Louis G. Birta—Ottawa Center of the McLeod Institute of Simulation Sciences
- Dr. Iva Smit, E&E Consultants, Netterden, The Netherlands.

The rationale for the code is clarified in:

Ören, T. (2002). Rationale for A Code of Professional Ethics for Simulationists. Proceedings of the 2002 Summer Computer Simulation Conference, pp. 428–433. <https://www.site.uottawa.ca/~oren/index-pubs/pubs-2000s.pdf>

The code is posted at different languages:

English https://scs.org/wp-content/uploads/2015/12/Simulationist-Code-of-Ethics_English.pdf

Turkish https://scs.org/wp-content/uploads/2015/12/Simulationist-Code-of-Ethics_Turkish.pdf

French https://scs.org/wp-content/uploads/2015/12/Simulationist-Code-of-Ethics_Turkish.pdf

Italian https://scs.org/wp-content/uploads/2015/12/Simulationist-Code-of-Ethics_Italian.pdf

Chinese https://scs.org/wp-content/uploads/2015/12/ZH-20150810-03-Code_0_Chinese_Zhang.pdf

Bulgarian https://scs.org/wp-content/uploads/2020/08/Simulationist-Code-of-Ethics_Bulgarian.pdf

The English version of the Code is provided here in the following paragraphs.



Simulationist Code of Ethics

Preamble

Simulationists are professionals involved in one or more of the following areas:
Modeling and simulation activities.

Providing modeling and simulation products.

Providing modeling and simulation services.

1. Personal Development and Profession

As a simulationist I will:

- 1.1 Acquire and maintain professional competence and attitude.
- 1.2 Treat fairly employees, clients, users, colleagues, and employers.
- 1.3 Encourage and support new entrants to the profession.
- 1.4 Support fellow practitioners and members of other professions who are engaged in modeling and simulation.
- 1.5 Assist colleagues to achieve reliable results.
- 1.6 Promote the reliable and credible use of modeling and simulation.
- 1.7 Promote the modeling and simulation profession; e.g., advance public knowledge and appreciation of modeling and simulation and clarify and counter false or misleading statements.

2. Professional Competence

As a simulationist I will:

- 2.1 Assure product and/or service quality by the use of proper methodologies and technologies.
- 2.2 Seek, utilize, and provide critical professional review.
- 2.3 Recommend and stipulate proper and achievable goals for any project.
- 2.4 Document simulation studies and/or systems comprehensibly and accurately to authorized parties.
- 2.5 Provide full disclosure of system design assumptions and known limitations and problems to authorized parties.
- 2.6 Be explicit and unequivocal about the conditions of applicability of specific models and associated simulation results.
- 2.7 Caution against acceptance of modeling and simulation results when there is insufficient evidence of thorough validation and verification.
- 2.8 Assure thorough and unbiased interpretations and evaluations of the results of modeling and simulation studies.

3. Trustworthiness

As a simulationist I will:

- 3.1 Be honest about any circumstances that might lead to conflict of interest.
- 3.2 Honor contracts, agreements, and assigned responsibilities and accountabilities.
- 3.3 Help develop an organizational environment that is supportive of ethical behavior.
- 3.4 Support studies which will not harm humans (current and future generations) as well as environment.

4. Property Rights and Due Credit

As a simulationist I will:

- 4.1 Give full acknowledgement to the contributions of others.
- 4.2 Give proper credit for intellectual property.
- 4.3 Honor property rights including copyrights and patents.
- 4.4 Honor privacy rights of individuals and organizations as well as confidentiality of the relevant data and knowledge.

5. Compliance with the Code

As a simulationist I will:

- 5.1 Adhere to this code and encourage other simulationists to adhere to it.
- 5.2 Treat violations of this code as inconsistent with being a simulationist.
- 5.3 Seek advice from professional colleagues when faced with an ethical dilemma in modeling and simulation activities.
- 5.4 Advise any professional society which supports this code of desirable updates.

References

1. Dancy J (1985) The role of imaginary cases in ethics. *Pac Philos Q* 66:141–153
2. Durán JM (2018) Computer simulations in science and engineering. In: *Ethics and computer simulation*. Springer, pp 171–188
3. Durán JM (2019) The ethics of climate simulations: the case of an F3 type tornado, forthcoming in *Science and the Art of Simulation* (Preprint: juanmduran.files.wordpress.com/2019/05/the-ethics-of-climate-simulations.pdf)
4. Williamson TJ (2010) Predicting building performance: the ethics of computer simulation. *Build Res Inf* 38(4):401–410
5. Brey P (2008) Ethics and emerging technologies. In: *Virtual reality and computer simulation*. Palgrave Macmillan, pp 315–332
6. Ören T, Elzas MS et al (2002) Code of professional ethics for simulationists. In: *Summer computer simulation conference*, Society for Computer Simulation International
7. Van de Poel I, Royackers L (2018) *Ethics, technology, and engineering: an introduction*. Wiley-Blackwell, Chichester, West Sussex, UK
8. Wallace WA (ed) (1994) *Ethics in modeling*. Emerald Group Publishing Ltd., Bingley, UK
9. Ören TI (2003) Rationale for a code of professional ethics for simulationists. In: Waite F et al William (ed) *Proceedings of 2002 summer simulation conference*, pp 428–433
10. Fowler HW (author), Crystal D (ed) (2010) *A dictionary of modern English usage: the classic*, 1st edn. Oxford University Press
11. Balebako R, O'Mahony A, Davis PK, Osoba OA (2019) Lessons from a workshop on ethical and privacy issues in social-behavioral research. In: Davis PK, O'Mahony A, Pfautz J (eds) *Social-behavioral modeling for complex systems*. Wiley, Hoboken, NJ, pp 49–61
12. Buolamwini J, Gebru T (2018) Bender shades: intersectional accuracy disparities in commercial gender classification. *Proc Mach Learn Res* 81:77–91
13. Kitroeff N (2019) Boeing 737 Max simulators are in high demand. They are flawed. *New York Times* May 17. Accessed at <https://www.nytimes.com/2019/05/17/business/boeing-737-max-simulators.html>
14. Federal Aviation Administration (2020) Preliminary summary of the FAA review of the Boeing 737 Max: return to service of the Boeing 737 MAX Aircraft, Version 1. <https://www.faa.gov/news/media/attachments/737-MAX-RTS-Preliminary-Summary-v-1.pdf>
15. Tolk A (2017) Code of ethics. In: Tolk A, Ören T (eds) *The profession of modeling and simulation: discipline, ethics, education, vocation, societies, and economics*. Wiley, Hoboken, NJ
16. McLeod J (1986) “But Mr. President—Is it ethical?” In: Wilson JR, Herniksen JO, Roberts SD (eds) *Proceedings of 11986 winter simulation conference*, pp 69–71

17. LeRon Shults F, Wildman WJ, Dignum V (2018) The ethics of computer modeling and simulation. In: Rabe M et al (eds) Proceedings of 2018 winter simulation conference, pp 4069–4083
18. Frankel MS (1989) Professional codes: why, how, and with what impact? *J Bus Ethics* 8 (2):109–115
19. Busert P (2018) Examine the professional codes of design organizations. In: Proceedings of design research society
20. Parker DB (1968) Rules of ethics in information processing. *Commun ACM* 11(3):198–201
21. Amy DJ (1984) Why policy analysis and ethics are incompatible. *J Policy Anal Manage* 3 (4):573–591
22. Wolf CW (1980) Ethics and policy analysis, P-6463-2. RAND Corp., Santa Monica, CA
23. Ben Veniste G (1983) On a code of ethics for policy experts. *J Policy Anal Manag* 3(4):561–572
24. Boston J, Bradstock A (2011) Public policy: why ethics matters. ANU E Press
25. Davis PK (2002) Analytic architecture for capabilities-based planning, mission-system analysis, and transformation, MR1513. RAND Corporation, Santa Monica, CA. Accessed at http://www.rand.org/pubs/monograph_reports/MR1513.html
26. Davis PK (2014) Analysis to inform defense planning despite austerity, RR-482. RAND Corporation, Santa Monica, CA. Accessed at http://www.rand.org/pubs/research_reports/RR582.html
27. Lempert RJ, Popper SW, Bankes SC (2003) Shaping the next one hundred years: new methods for quantitative long-term policy analysis. RAND Corporation, Santa Monica, CA. Accessed at http://www.rand.org/pubs/monograph_reports/MR1626.html
28. Marchau VAWJ, Walker WE, Bloemen PJT, Popper SW (eds) (2019) Decision making under deep uncertainty: from theory to practice. Springer, Cham, Switzerland
29. Davis PK, O’Mahony A, Pfautz J (eds) (2019) Social-behavioral modeling for complex systems. Wiley, Hoboken, NJ



Enterprise Modeling and Simulation

9

Hezam Haidar , Nicolas Daclin , Gregory Zacharewicz ,
and Guy Doumeingts 

Abstract

Manufacturing and other industries are entering the digital age with all its challenges, from systems-of-systems constraints to interoperability challenges. This chapter of the SCS M&S Body of Knowledge discusses challenges that can be addressed by enterprise modeling and simulation. Such challenges include collaboration with partners, establishing supply chains, and enterprise optimization without becoming too brittle in an agile environment are.

Keywords

Modeling and simulation · Enterprise modeling · Graph with Results and Activities Interrelated (GRAI) · Business Process Modeling and Notation (BPMN) · Model Driven Interoperability System Engineering (MDISE)

H. Haidar (✉)
INTEROP-VLap, Brussels, Belgium
e-mail: hezam.haidar@interop-vlab.eu

N. Daclin · G. Zacharewicz
IMT-Mines Ales, Alès, France
e-mail: nicolas.daclin@mines-ales.fr

G. Zacharewicz
e-mail: Gregory.Zacharewicz@mines-ales.fr

G. Doumeingts
Bordeaux University, Nouvelle-Aquitaine, France
e-mail: guy.doumeingts@ims-bordeaux.fr

9.1 Introduction

Hezam Haidar, Nicolas Daclin, Gregory Zacharewicz, Guy Doumeingts.

Traditional manufacturing companies are entering the digital age either internally or when they need to collaborate [1]. The Information and Communication Technology (ICT) sector is faced with an increasing amount of information exchanged between partners through machines (physical means), people/organization, and IT in the context of business collaboration. Interoperability management is becoming increasingly critical, but it is not yet fully anticipated, controlled, and effectively supported to recover from security problems or failures.

Enterprises decision-makers are faced by several questions when collaboration with partners within a supply chain process is required. Based on our experience in enterprise and business modeling on which we accompany companies in their projects, many questions arise. The most frequently received questions from companies are: What is the main objective of the collaboration? How to organize the collaboration? What interoperability barriers must be to overcome? What about focusing on the interaction with actors and humans? These questions clearly list the need of guidelines, methodology, and simulation support.

This chapter intends to propose a model-driven method that addresses simulation in existing model-driven methods. For that purpose, it elaborates the Model-Driven Interoperability System Engineering (MDISE) that focuses on the vertical and horizontal interoperability model-driven approach between enterprises while MDSEA remains focused on enterprise integration between internal domains (IT, human/organization, physical means) before connecting the different models. The chapter concludes with some current development of the MDISE framework and method with model system tool box (MSTB) that evolved in the frame of Interop-V-Lab Task force. Finally, it gives some perspectives about the interest of MDISE in the frame of future cyber-physical system (CPS) research works.

9.1.1 Problem Statement About Enterprise Modeling and Simulation

An enterprise is an organization composed of people, activities, information, and resources involved in supplying a product or service to a consumer [2]. Physical supply chain activities involve the transformation of natural resources, raw materials, and components into a finished product that is delivered to the end customer [3]. This work focuses on enterprise system (ICT Supply Chain 2020), which requires the management of data linked by computer components. In addition, on each link between ICT components, different types of resources are also involved, so different simulation problems can arise. In the frame of Industry 4.0, a cyber-physical system (CPS) [4] and its environment can be considered as relevant instances of SC-ICTS with the inherent need of simulation.

According to common definitions, supply chain management (SCM) is the management of the flow of goods and services and involves the movement and

storage of raw materials, work-in-process, and finished goods from the point of origin to the point of consumption. Here, we consider interdependent networks of goods or services, where ICT supply chain management is required to manage the channels and nodes for the delivery from source to end customers. To support services, SC-ICTS simulation is widely recognized as a major concern for organizations (ICT Supply Chain 2020) and companies [5]. More technically, SC-ICTS refers to data/information exchanges among ICT systems involved in physical supply chains or industrial systems. For instance, [6] have defined an ICT supply chain as “the full set of actors included in the network infrastructure”. It includes end-users, policy makers, procurement specialists, systems integrators, network provider, and software/hardware vendors that produce (big) data.

While they are booming, these systems face conceptual and technological barriers that can limit their adaption. The lack of simulation is the cumulative effect of the increased sophistication of ICT, the scale of the information systems, and the increasing speed and complexity of a distributed global supply chain. The lack of sufficient visibility and control throughout the ICT supply chain is making it increasingly difficult to understand the exposure of the enterprise and manage the simulation associated with the supply chain. This, in turn, increases the risk of miss-exploiting the supply chain through a variety of means, including materials, products, data, and cyber-physical resources and processes.

The authors in Reference [7] identified a demand for supply chain simulation guidance. However, the ICT supply chain discipline is in an early stage of development with diverse perspectives on foundational SC-ICTS definitions and scope, disparate bodies of knowledge, and fragmented standards and best practice efforts. Additionally, there is a need to identify the available and needed tools, technology, and research related to ICT supply chain simulation and better understand their benefits and limitations.

In brief, the SC-ICTS is not yet fully standardized or even well-defined. Yet, potential supply chain participants attempt to find or define terms, definitions, characterizations of the collaboration, but frequently fail to identify and evaluate current and SC-ICTS-related standards and practices (need, scope, and development approach). In consequence, a methodology that list models, tools, technology, and techniques useful in securing the building of ICT supply chain is still wanted. For that purpose, this chapter will acclaim to join efforts with methodology to improve the efficiency of SC-ICTS simulation based on a model and an approach to answer Industry 4.0 needs due to the hybrid/heterogeneous composition of CPS, they are interesting candidate nodes for this SC-ICTS approach.

9.1.2 Methodological and Technical Approach

According to the objective of identifying a list of models, tools, technology, and techniques useful in building consistent and interoperable ICT supply chain, this section recalls components about enterprise modeling, simulation, and MDSEA, which contribute to building a model-driven simulation for systems.

9.2 Enterprise Modeling

Hezam Haidar, Nicolas Daclin, Gregory Zacharewicz, Guy Doumeingts.

Enterprise modeling (EM) is the abstract representation, of an enterprise with its structure, the various functions, the processes, the information, the resources (physical and human), the management, the relationships with the environment (customers and suppliers), and all activities required to produce industrial products or services. The goal of EM is to represent (based on models) a system as it stands and improve its performances or to follow the evolution of the enterprise. Additionally, the relation of EM and integration domain has been considered [8].

Enterprise modeling concepts in industrial environment were developed, starting at the end of 1970's, mainly in USA by the Department of Defense (DoD), in order to improve the competitiveness of the industry that seems at this period to be behind the competitiveness of the Japanese industry. A second reason was the more and more use of Information Technology (IT) in manufacturing and the appearance of a new way to design manufacturing systems: computer-integrated manufacturing (CIM). The DoD launched several projects in cooperation with industrial companies such as Boeing, General Electric, Westinghouse, IBM, Hughes Aircraft, and Softech Inc. One of the first formalisms developed to represent a part of EM concept in this new approach was the IDEF method (integrated definition) (IDEX) [9], for which a series of formalisms were proposed. Among them: IDEF0 to represent functions and activities with a simple syntax and a hierarchical decomposition from a global representation of the enterprises to a detailed representation, IDEF1 to represent information, and IDEF 3 to represent the logic of process execution, which can be used to develop a simulation tool.

At the same time, in Europe, the Group of Research in Automation Integration (GRAI) of the University of Bordeaux developed the graph with results and activities interrelated (GRAI) and also the GRAI model [10] to represent the manufacturing based on system theory [11, 12, 13], the theory of hierarchical multilevel system [14], which allows the decentralizing of the decision-making and to increase the reactivity, the organization theory [15, 16], the discrete event systems [17, 18], and the production management concepts [19, 20]. Three subsystems are defined: physical (Fig. 9.1 (transformation of purchased items and information in products or services)), decisional (to control the physical system (Fig. 9.1)), and information (to manage the creation and the exchange of information (Fig. 9.2)). This research work was completed by a cooperation with the industry to validate the concepts: Télé-mécanique Electrique (in Nice) and Merlin Gerin (in Grenoble (today both in Schneider Electric), and Crouzet (in Valence) in order to improve the performances of workshops; SNECMA (today Safran) in Le Creusot to design a flexible manufacturing system (FMS), AMRI (near Bordeaux) to design a FMS, and other companies such as Suez to improve the management of water distribution and Airbus Toulouse to improve the performance of a composite workshop. In the last four years, the GRAI model and method have been extended to be applied in the domain of services, but also to develop integrated solutions in the three domains: Information Technology

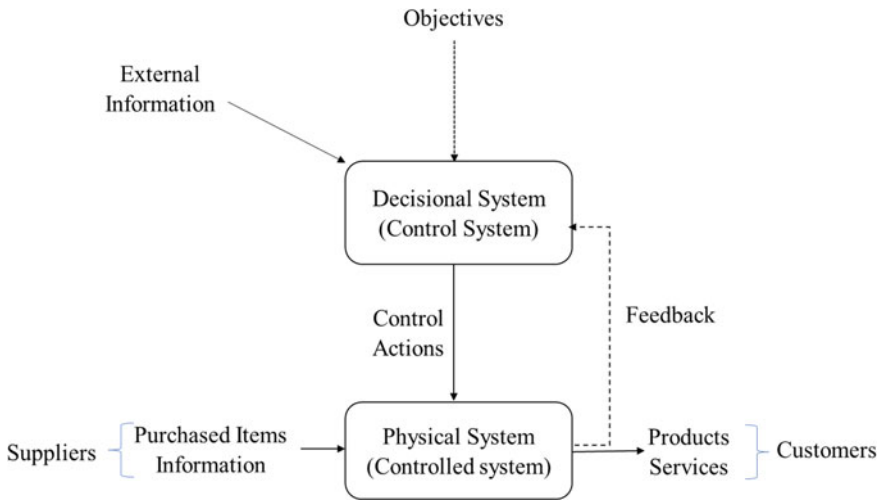


Fig. 9.1 Physical system and the control system

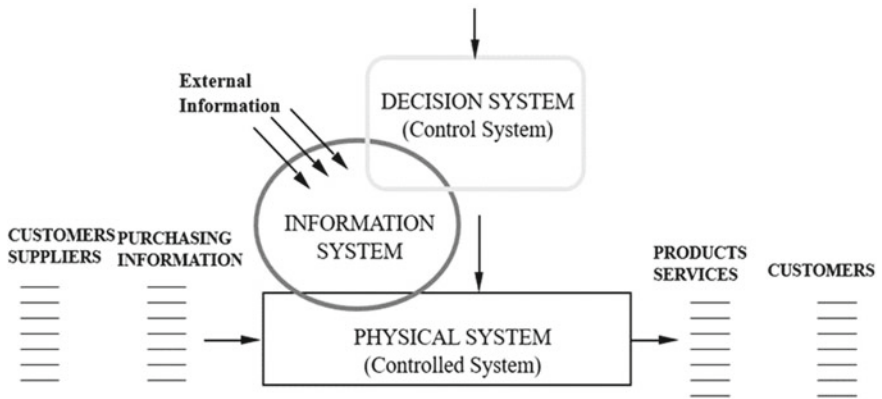


Fig. 9.2 Creation of the information system in the GRAI model

(IT), Physical System, and Organization/Human System called Model-Driven System Engineering Architecture (MDSEA (see Sect. 3.3.4). At the same time, other methods appear, one major one is CIMOSA [21], which was developed in the late 1980's. Additionally, IEM [22] and ARIS [23] have been largely used.

9.2.1 GRAI Model and GRAI Formalisms

The previous section focused on the main theories that have supported the creation of the GRAI model. This section describes the structure of the basic model and the

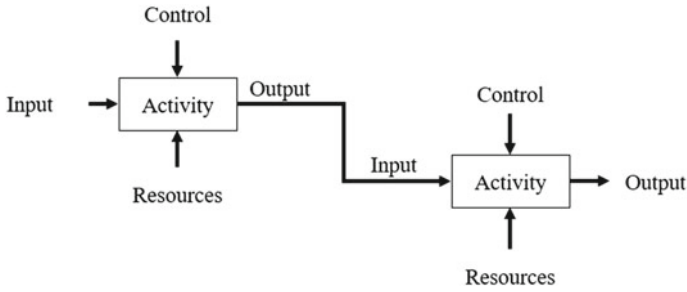


Fig. 9.3 IDEF0

formalisms to describe the enterprise. The previous concepts allow us to consider the enterprise as a complex system that can be split up into two entities (Fig. 9.1): The physical system or controlled system (also called the transformation system) which produces the products or/and the services, the decisional system (control system) that controls the physical system.

This systemic view introduces the concept of control loop. In Fig. 9.2, the information system is added to manage all the information.

Currently, GRAI model uses various formalisms to graphically represent the components of a manufacturing system: physical, decision, and information.

Concerning the modeling of activities, two formalisms are selected: IDEF0 and extended actigram star: (EA*). In IDEF0 (Fig. 9.3), there are four types of flows:

- Input represents the flow of entities which will be transformed.
- Output represents the flow of entities which have been transformed.
- Control represents the conditions or circumstances that govern the transformation (objectives, constraints ...).
- Resource represents the resources required to perform the transformation.

Extended actigram star (EA*) formalism is in line with IDEF0 and IDEF3 to facilitate the transformation of models from bottom business specific model (BSM) level to technology independent model (TIM) level [24]. The other GRAI formalisms are

- Global level for the control using GRAI grid formalism (Fig. 9.4)
- Detailed level for the control using GRAI nets formalism derived from EA* (Fig. 9.5).

The GRAI grid is a formalism which represents the decisional subsystem. It is a matrix in which functions, decision levels, decision centers, and decision links are identified as follow.

The functions are represented vertically; a function includes a set of activities that contributes to the same purpose. The decision levels for these functions are represented horizontally and define the temporality of the decisions. The criteria of

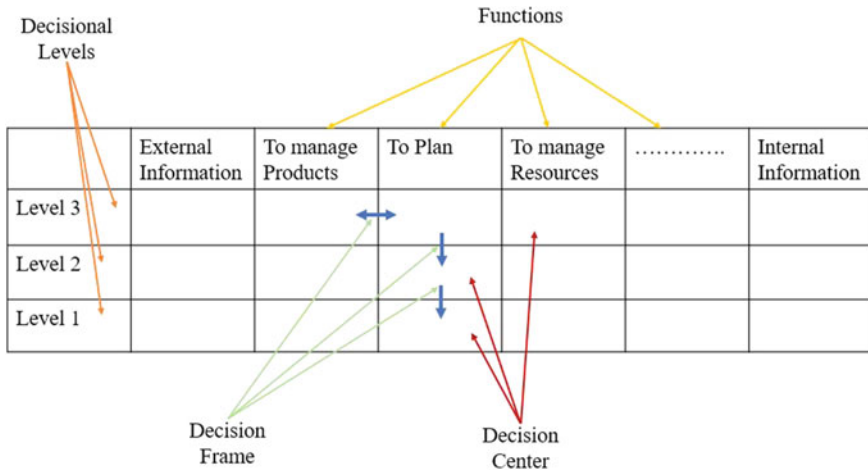
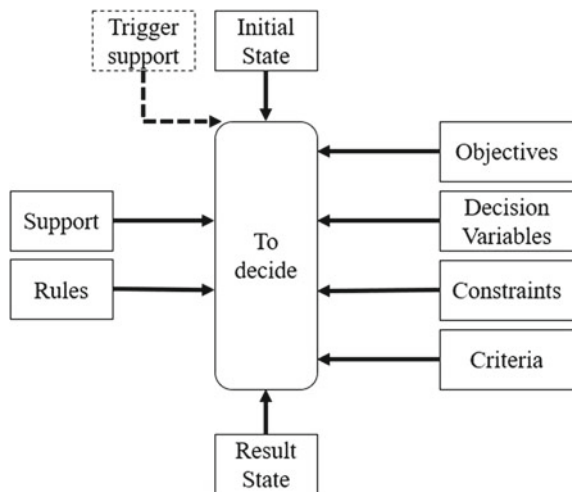


Fig. 9.4 GRAI grid formalism

Fig. 9.5 GRAI nets formalism

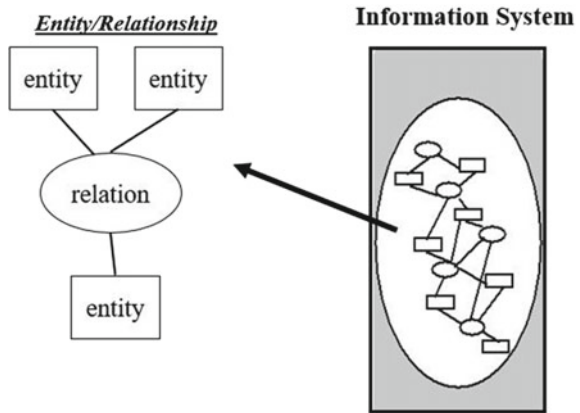


decomposition are the horizon and the period of time. Each cell represents a decision center, i.e., intersection between a function and a decision level.

The decision frames represent the hierarchical links between decisions and include all information for decision-making (objective, decision variable, constraint, and criteria).

GRAI nets (Fig. 9.5) give the detailed description of the various activities in each decision center identified in the GRAI grid. By using GRAI nets, the result of one discrete activity can relate to the support of another discrete activity. With GRAI nets, four fundamental elements are to be identified:

Fig. 9.6 Information system formalism



- To do or to decide (activity name),
- Initial state (main input of an activity),
- Supports (information, decision frame, methods, and materials),
- Results (results of an activity).

The formalism used to describe the information system is entity/relationship modeling proposed by UML (Fig. 9.6). It describes the information structure in coherence with the decisional system.

Several IT tools have been developed to support the description of formalisms. The last one is the model system tool box (MSTB), as described in Sect. 5.1.

9.2.2 BPMN

At more technical level, business process modeling and notation (BPMN 2.0) (business process model and notation [25] language can be used. It is more complex to use than EA* but offers a wider range of detailed process modeling concepts. It is formalized in XML format, making model transformation easy. In addition, BPMN allows the representation of human and technical resources that are required in model-driven approaches representation principles. BPMN has the advantage of providing a meta-model developed by the object management group (OMG) that facilitates its implementation. Finally, it prepares the transition to the lower levels on the IT aspect thanks to its simulation with many BPM IT platforms, thus allowing the deployment and semi-automatic transformation for the execution of BPMN processes.

9.2.3 Other Formalisms for Information System Design

With a more technical view of information systems than BPMN, the open group architecture framework (TOGAF) and architecture-animate (ArchiMate) models can be used to capture other views at a more technical level. [26, 27]. In details, the

enterprise architecture frameworks TOGAF and ArchiMate propose different layers from business level to application level to design the information system of organization. TOGAF with its ADM cycle highlights a step-by-step methodology to migrate toward a new information system consistently. It does not propose any languages and relies on existing ones and adapted such as UML. ArchiMate proposes different models at each layer (motivation, business, application, and technology) in addition to its framework. Let us note that the ArchiMate specification can be used with TOGAF to build expected models. While the languages proposed and deployed in these frameworks are fully adapted to develop an information system that meet enterprise expectations, they allow for the representations of different points of view but often in a less accurate way than a language fully dedicated and developed for a particular point of view. Some points of view are not considered by existing frameworks such as, for instance, the decisional and physical points of view. In addition, dedicated languages often go beyond the descriptive aspect and propose means to analyze and improve the system under study. This is the case, for instance, with the GRAI methodology that proposes formalisms (GRAI grid and GRAI networks) to model and analyze the decisional point of view of an organization.

9.2.4 Conclusions

Currently, EM is not used as expected in industrial world, particularly in Europe. It seems that in USA, the use is more important, certainly based on the influence of IDEFx. Education must be developed in this domain by elaborating examples based on the concrete experience with real cases. Another argument is the development of end-users-oriented and adapted IT tools because they capture the knowledge on their own manufacturing system. For this purpose, the graphical modeling aspect and ease of use are very important. The last objective is to link EM to other areas like enterprise simulation and the model-driven approach as proposed in Sect. 3.5. Modeling Enterprise at the Different Levels of Abstraction

Based on the modeling levels just previously described, the methodology MDSEA proposed to associate relevant modeling languages at each level to represent confidently the existing system and the future service product and service system. To achieve this goal, the standards for process modeling are gaining importance, with several process modeling languages and tools available to enhance the representation of enterprise processes. To choose among the languages, the level of abstraction required is important.

The first specification step of a model to be established between two partners is crucial. At the BSM level, the modeling language must be simple to use, expressive, and understandable by business-oriented users. Moreover, this (or these) language(s) must cover processes and decisions with coherent models. The choice is affected by the capacity of the language to propose a hierarchical decomposition (global view to detailed ones), which is especially required at this level. Indeed, business decision-makers often have a global view of the running system and need

languages allowing this global representation with few high-level activities (physical process or decisional activities). This global view must be completed by more detailed activities models elaborated by the enterprise sector responsible. These models are connected to top level models in a hierarchical and inclusive way. These are the principles of systemic and system theory to consider selecting the languages. However, it is also obvious that the choice of modeling languages is subjective, depending on the experience of the languages' practitioners and on their wide dissemination within enterprises.

As for process modeling at the business level (BSM), several languages exist. Extended actigram* (EA*) presented in Sect. 3.1 was chosen to model processes at the BSM level due to its independence regarding IT consideration, its hierarchical decomposition, and the fact that it can model three supported resources: material, human/organization, and IT. It has been developed as an answer to the previous issues encountered with IDEF0 regarding its simulation with BPMN for example. It intends to capture business process models at a high-semantic level, independently from any technological or detailed specifications. Service-oriented modeling and architecture principles [28], developed by IBM, were also considered, but these languages are more IT oriented and thus were far away from our industrial requirements.

At the TIM level, BPMN 2.0 is used because this language offers a large set of detailed modeling constructs, including IT aspects and benefits from the simulation of many BPMN? IT platforms allowing for the deployment and automated transformation for the execution of BPMN processes. Moreover, BPMN also enables the representation of human and technical resources, which are required in the MDSEA principles of representation. BPMN also has the advantage to provide a metamodel developed by OMG, which facilitates the implementation of the language. It is also extensible with third party metamodels, which is important and respects the OMG simulation standards (e.g., Xmi).

In detail, GRAI approach is to be used by business representatives at BSM and BPMN at the TIM level. BPMN is used to be the backbone language between the business view and IT level. However, because the languages have different consideration and view on the system, it must be able to link them. In detail, the EA* models designed at BSM level need to be transformed into BPMN 2.0 models to obtain the coherent business process models at the TIM level.

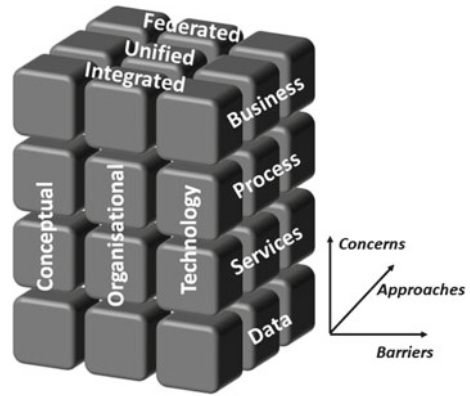
9.3 Driving Models to Simulation

Hezam Haidar, Nicolas Daclin, Gregory Zacharewicz, Guy Doumeings.

9.3.1 Interoperability

According to ISO 11354, enterprise interoperability is the “ability of enterprises and entities within those enterprises to communicate and interact effectively”. To structure the different concept of interoperability, ISO 11354 makes available a

Fig. 9.7 Framework for enterprise interoperability



framework that provides a set of interoperability solutions relevant with the practitioners' requirements. Thus, this framework for enterprise interoperability relies onto three dimensions such as concerns, barriers, and approaches (Fig. 9.7).

Interoperability concerns highlights the interoperability viewpoints, i.e., the levels in enterprises at which interoperability needs to be developed. Interoperability concerns include business level (ex. working methods, decision-making...), process level (collaborative business processes), service level (application deployed in collaborative processes), and data (shared and exchanged within the process and through application).

Interoperability approaches take the classical approaches proposed in ISO 14258: integration unification and federation. Integration encourages the use of a common format through all collaborative organizations (ex. use of BPMN 2.0 language to model processes). Unification relies on the use of a "meta"-level principles to ensure the mapping between different formats (ex. use of model-driven engineering approach). Federation promotes to develop mechanisms allowing to collaborative organization get used to each other's methods, data, and tools on the fly (no use of standard or any mapping).

The barriers represent the problems of interoperability that can occur between the organizations. Conceptual barrier deals with exchanged information (syntax and semantic problems) [29]. Technological barrier deals with the compatibilities issues between application and information systems. Lastly, organizational barrier deals with the definition of responsibilities and authorization of involved actors, authority, process, and regulatory aspects.

The intersection of three dimensions (e.g., conceptual x process x unification) makes available a set of relevant solution to develop interoperability according to the intersection's requirements. Thus, the integrated approach, despite it is constraining, is likely the easy way to set up interoperability since each organization adopts the same methods, models, or tools. The unified approach seems the most implemented approach since the concepts and tools are well identified, defined, and equipped, the model-driven engineering or else model-driven engineering and their

practices are the most known approaches. Lastly, the federated, although it represents the most challenging approach and meets the simulation “spirit” expectations (no mapping, no standards but a dynamic and continuous adaptation), still remain poorly developed. Thus, the Enterprise Simulation roadmap published by the European Commission [30], developing the federated approach for interoperability is considered to be one of the research challenges in the next years.

9.3.2 Vertical Decomposition: Toward Alignment from Business to Operational

Considering resource domains while modeling at the bottom BSM helps to anticipate how the different kinds of resources will be called, how they will interact with the other components of the system and how they will be used to perform the process. Nevertheless, it requires an extraction strategy by choosing appropriate methods and models to get their specificity properly.

Figure 9.8 shows the interest of such architecture that is to design and implement a service product and to produce a dedicated service system coherent with business service models, represented with enterprise models. Looking at TIM and TSM levels show how the methodology differentiates three kinds of resources categorized into IT, human, and physical means. The reason is to tackle the different requirements of resources at the implementation stage of the service system. Then, the implementation of the resources detailed in the TSM model allows for the implementation of the service system and related service product through a set of services, i.e., a system in which the service provider (an enterprise inside a network, or in a cloud of service providers) is not directly identified by the customer, which

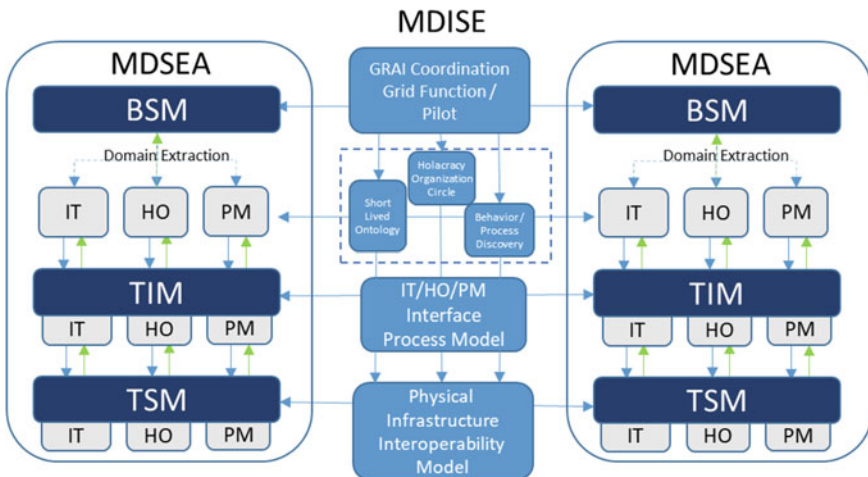


Fig. 9.8 MDISE architecture for enterprise interoperability

can only remain interfaced with a service delivery. The service maintenance and decommission activities can be ensured by different companies in the network without direct identification by the customer. However, the virtual organization keeps the property rights on the services.

About IT domain, several model languages exist. GRAI introduced at the beginning of the chapter has demonstrated the capacity to tackle modeling aspect from the decisional perspective at the BSM level. At the lower level, UML can be used to describe more technical views.

About physical means, some physical models can help to better catch the behavior of machines used in the systems. It can include performance models as well as other expressed properties thanks to physical and mathematical models to be considered in this part of the model. This topic is being discussed in several simulation projects (I-V-Lab (<http://interop-vlab.eu/projects-i-vlab/>)), including the DIH4CPS project [31].

About human and organization, we believe that holacracy, which is decision-making distributed throughout a holarchy of self-organizing teams, can bring people to work together. The challenge is to catch and model holacracy systems.

It is important to mention that the service system represented at each level of MDISE remains the same system, but with details and including implementation constraints. Nevertheless, after having described each category of resource with appropriate models, another challenge is to deal with the coupling of these models together. For this aim, simulation plays the role of gluing them together.

Additionally, in Sect. 5.1, MDISE vertical decomposition will be implemented in MSTB evolved as an open-source tool extended to cover new category of models introduced in the next section. The new level of description introduced here will be considered as well, such as decisional models in addition to process models and human machine interaction in the simulation management life cycle. The service approach will keep driving this development [32].

9.3.3 Horizontal Alignment: Toward Simulation for Better Collaboration Between Service Network

Figure 9.8 shows the collaboration between two enterprises to produce a service. Collaboration between different entities can happen at different MDSEA abstraction levels (BSM, TIM, and TSM). The BSM models allow to represent the TO BE models of both entities and to align the simulation of practices in terms of business process models and decision models. In MDSEA, simulation is a key factor for enterprise collaboration. Enterprise models ensure not only simulation of practices, but also between the human resources and IT systems supporting these practices.

Business Service Model (BSM): BSM specifies the models, at the global level, describing the service running inside a single enterprise or inside a set of enterprises as well as the links representing cooperation between these enterprises. The models at the BSM level must be independent of the future technologies that will be used for the various resources and must reflect the business perspective of the service

system. In this sense, it is useful, not only as an aid to understand a problem, but also it plays an important role in bridging the gap between domain experts and the development experts who will build the service system. The BSM level allows for the defining of the link between the production of products and the production of services.

Technology Independent Model (TIM): TIM delivers models at a second level of abstraction independent from the technology used to implement the system. It gives detailed specifications of the structure and functionality of the service system, which do not include technological details. More concretely, it focuses on the operational details while hiding specific details of any technology to stay independent from any technology, used for the implementation. At TIM level, the detailed specification of a service system's components is elaborated with respect to IT, organization/human, and physical means involved within the production of the service. It is important to mention that, in comparison with MDA or MDI or service-oriented modeling and architecture (SOMA), the objective of MDSEA is not only IT oriented, and then, this requires enabling the representation of human and technical resources from the BSM level. At the TIM level, the representations must add some information in comparison with the BSM models.

Technology Specific Model (TSM): TSM enhances the TIM model specifications with the implementation details of the system, i.e., how it will use a specific technology or physical means (IT applications, machine, or a person) for delivering services in the interaction with customers. At TSM level, the models must provide sufficient details to develop software applications, hardware components, recruiting human operators/managers or establishing internal training plans, buying, and realizing machine devices. As for IT applications, a TSM model enhances a TIM model with technological details and implementation constructs that are available in a specific implementation platform, including middleware, operating systems, and programming languages (e.g., Java, C++, EJB, CORBA, XML, Web Services, etc.). After the technical specifications given at TSM level, the next step consists in the implementation of the service system in terms of IT components (applications and services), physical means (machine or device components or material handling), and human resources and organization ensuring human related tasks/operations.

Initially, the simulation models developed in the MDI focus on the principles of "mappings" to establish interoperability. In that sense, it implements the unified approach and requires the linking of concepts and relations of heterogeneous modeling languages, for example. This kind of approach is robust but time consuming, with a possibility of a partial overlapping of languages (e.g., one concept does not exist in both) requiring the extension of the languages and to develop transformation rules that can change if languages change.

Thus, this approach is completely relevant, especially for collaborative organization mid- and long-term-oriented, i.e., stable over time and for which the intensity of the collaboration tends toward cooperation and collaboration and an important level of integration, according to Reference [33].

In the frame of MDISE, the purpose is to extend the MDSEA and MDI principles to a federative approach to develop simulation. This means to prevent, as

much as possible, any common format or predetermined model, and each partner keeps its own organizational structure, business processes, tools, data format, etc.

To this end, the goal is to create a simulation model to insert between organizations. This model aims to identify and allow simulation independently of the models, organizational structure, or physical means used by partners. This model can be initiated with known and simple but limited mappings (or any other basic mechanisms) to avoid reaching a unified or integrated approach. Thus, it must be built on the knowledge about the characteristic of partners without any (or at least strictly limited) modification or adaptation. These characteristics are the interfaces (I/O) requested for the collaboration (functional and/or physical), human resources, data, models, etc., allowing for the establishment of consistent interaction. Thus, the proposed simulation model does not take any interest in the modeling language, organizational structure, or physical means used by partners and does not aim to establish a strict mapping or equivalence between them. It aims to build a transient simulation bridge based on the identification and the analysis of knowledge, constraints, and specific features stemming from partners. It should be noted that the principle to build a “centric simulation model” approach to the “mutual adjustment”, mentioned in Reference [34], thus fits the federated approach of the simulation framework.

Therefore, whether for the IT, the human/organization, or the physical means domain, this model can be considered in two ways:

- A “component mode” relying on commercial off-the-shelf (COTS) sufficiently generic to be deployed in different organizations. These bricks are pre-existing basic models (or skeleton) from identified and known simulation situations. These atomic COTS belong to a set and can be combined to provide a complex COTS to establish simulation in specific situations. They cannot be modified and are used from identified characteristics and requirements of the collaboration such as the synchronization, integrity, quality, or quantity of data. For instance, the buffer is a well-known mechanism that can be used for the IT domain to allow a synchronization between two processes.
- An “emergent” mode relying on a model built on the fly for complex requirements and constraints making the direct use of “component mode” impossible. In this case, the model is based on rules allowing for the building of simulation from scratch. These rules are built from the specificities of the collaboration in terms of IT, organization, or physical means. These primo rules set can be raised with other discovered rules. In that sense, the use of techniques from artificial intelligence (self-learning, process mining, data mining, etc.) is an important challenge for this approach. Moreover, a simulation model highlighted in the emergent mode can become a COTS in the component mode if it appears regular in different collaborative organizations. Thus, the purpose of this mode is to be free from any components—once a component deployed for simulation it cannot consider

modification of organization—and make a dynamic adaption possible in the case of the modification of partners and entailed constraints on the collaboration. For instance, the short-lived ontology can be used for the simulation of data in the IT domain, it uses an ontology valid for a limited duration. At the human/organizational level, the principles of the holacracy and its concepts of circles and roles can be considered, by way of an adaption for the simulation purpose, to make different organizational structures (hierarchical, functional, matrix, etc.) interoperable. Thus, by identifying actors from both sides, the definition of rules could authorize the building of time bounded circles and allowing for a coherent interaction between persons without any modification of internal structures. For the physical means domain, take the example of a floppy disk. The principles are to build a set of data that physically describe the system. The description of the object can be based on physical data (e.g., dimension), which is data related to the business or stemming from an image analysis. From this step, other partners can anticipate the reception of the object and be prepared to exploit it.

Lastly, both modes, “component” and “emergent”, can be used in a complementary manner. The simulation model can be initiated with existing components and continued with emergent ones if requested.

9.4 Implementing Framework and Method in MSTB Evolved

Hezam Haidar, Nicolas Daclin, Gregory Zacharewicz, Guy Doumeingts.

As an historical perspective, to operationalize the models from BSM to TSM, [35], Gregory [36] proposed the frame of the EU FP7 Research Project MSEE “Manufacturing Service Ecosystem” ID 284860 (<http://www.msee-ip.eu/>). The authors of References [35] introduced the implementation of the SLMToolBox that is a service graphical modeler, model transformer, and simulation engine. Since then, SLMToolBox has been improved and renamed MSTB. This tool has been implemented as an Eclipse RCP service. In detail, it runs the transformation from service processes models designed by business users to BPMN models. Then, the BPMN models are transformed into DEVS models to simulate the behavior of the entire process model. Thus, MSTB aims at proposing a TO BE process-oriented modeling framework to represent the architecture and data workflows that exist in the ICT supply chain at the TIM level of MDSEA.

Therefore, to meet the expectation expressed in the chapter, an operationalization of MDISE to extend MSTB according to the MDISE methodology is under development. This will allow for the identification and modeling of the enterprise frontier that can be initially poorly compatible with the environment and potentially places the interoperability barriers in organizational relations, including managing multi-tenancy, multi-providers, and system/service continuity. In addition, it will

make a methodology available to model the planning and execution to mitigate or avoid interoperability issues during the whole life cycle, such as considering the evolution of ICT from both IT and OT points of views. Models will identify and highlight the need for simulation. It will help users mitigate barriers to simulation using models and simulations to manage exceptions and ensure business continuity. The objective is to prevent an ICT simulation issues occurring during production or manage it with short business resilience duration. The new version of MSTB is called MSTB evolved.

9.4.1 Models and Model Transformation in MSTB (BSM Level)

To show the usability and applicability of MDISE and MSTB evolved in SC-ICTS, the methodology is detailed in this subsection. First, the conceptual workflows from the requirements established at level BSM are defined. Then, it prepares the technical works for the implementation of the information system.

9.4.2 Using GRAI Grid and Extended Actigram* at Top BSM

Among the different systems, complex systems (systems of systems and eco systems), and organizations, the GRAI grid focuses on modeling the decisional aspects of the management. The proposition in MDISE is to use the GRAI grid at the top of the BSM to define the coordination and simulation of two enterprises, detailing the points where decisions can be made (decision centers) while participating and the information relationships among these. In the frame of MDISE, models built using the grid allows for the analysis and design of how decisions are coordinated and synchronized at the frontier of two enterprises.

As for process modeling at the business level (top BSM), several languages exist. EA*, introduced in Sect. 3.1.1, is chosen to model processes at the BSM level due to its independence regarding IT consideration, its hierarchical decomposition, and the fact that it can easily model three supported resources: material, human and IT. It was developed as an answer to the previous issues encountered with other enterprise modeling languages regarding its capacity to represent interoperability [35]. However, EA* is chosen to capture business process models at a high-semantic level, independently from any technological or detailed specifications in MDISE. Service-oriented modeling and architecture principles [37] developed by IBM were also considered, but these languages are more IT oriented and thus were far away from our requirements. EA* provide at top BSM a common and explicit graphical notation for business process modeling of enterprises interfaces within MDISE, so it fits business-oriented people requirements, who need to describe and communicate high-level business processes involving enterprise resources with the help of a simple and explicit formalism. In comparison with other initiatives such as BPMN2.0, it relies on a reduce set of graphical objects and

focus on the “business” aspects of enterprise processes. The accessible syntax of EA* facilitates the design of business process.

To recap, at the top of BSM in MDISE, GRAI grid and EA* facilitate the modeling of business process and decision at the interface of the enterprise with its environment, offering a scalable view of the decision and process modeled. This level is addressed to users responsible of the creation of the first model, business people responsible of the management, and to technical developers responsible of the development of business process modeling tools. As a graphical modeling language, EA* and GRAI grid provide business users and analysts standards to visualize business processes in an enterprise, and thus in a comprehensible way.

9.4.3 Domain Specific Languages at Bottom BSM

At the bottom BSM, the approach needs to identify and catch different concepts related to the domains: IT, human, and physical means. To capture these concepts, models can facilitate description and abstraction. However, it is required to keep a simple set of modeling notations comprehensible by business users. This methodology will drive the BSM concepts down to TIM still independently of technologies. The proposition provides models to express each domain. Even at BSM, models will have to consider input/output information coming from the workflow along the supply chain. To support stakeholders, this methodology will make a library of potential simulation solutions available to handle them; they will be used to stress the models and simulate interoperability management scenarios to evaluate their interest.

According to Sect. 4, and at this MDISE stage, it is required to integrate domain specific models with a process-oriented way for each domain human, IT, and physical means:

At collaboration time, no orchestration is formalized between participant of two distinct entities and without any organizational structure between the enterprises. The idea of MDISE is to better train and support humans in this situation to reach a better response time in critical situations. The proposition takes advantage of holacracy structures and rules. Holacracy rules must be described by models. These models will provide a framework to help to customize the specific processes need for business process simulation. The holacracy consists of four key tools: 1. rationale, 2. role, 3. tension, and 4. meeting formats. These tools can be described with GRAI Net models introduced in Sect. 3.1.1.

Each data used by stakeholders have specific structure that leads to semantic issues. The use of a short-lived ontology concept [38] can tackle this barrier. Short-lived ontology fits the federated Enterprise Simulation approach highlighted in the EIF. It uses no common persistent ontology; the communication must be accommodated on the fly. In consequence, the ontology that structures the messages exchanged must be short-lived, (i.e., non-persistent). EA* diagram from GRAI can be used with the notation of the ontology validity period and eventually rules to set and modify the validity.

Finally, the physical means interaction processes that happen at the frontier between enterprises can be learned from good practices established in the past, in similar situations. Here, GRAI EA* models can be obtained from the process discovery approach to reveal interesting behavior from the legacy practice. For instance, process mining is an automated, data-driven AI technology that finds maps and documents of existing businesses tasks thanks to existing data.

9.4.4 Interface Process Model at TIM Level

This subsection focuses on the modeling of data workflows at the TIM level of MDISE. This task will select accurate language to the TIM level of modeling. These languages might be potentially specialized to clearly represent the data exchange. These data workflows will be derived thanks to the ATL model transformation from the bottom BSM conceptual models of Sect. 5.1.1. They will describe the data circulating from an operative level of ICT up to the decision department, as well as outside the enterprise along with other enterprise partners. The appropriate modeling language will allow for describing after the domain extractions of Sect. 4 data, handled both by human/organization with user devices, smart machines, and IT with M2M, at the technological independent level. It will also propose a methodology to transform these models inherited from the bottom BSM models proposed in Sect. 3.1. BPMN appears to be the most appropriate language due to its expressiveness, user-friendly description, and large user community. It would be the basis, enriched with specific concepts related to data security.

According to the partners' experiences and literature, the most appropriate domain patterns can be defined here. Among them, at the TIM level, BPMN 2.0 (introduced in Sect. 3.1.2) can be chosen to model the connection of the domains because it offers a large set of detailed modeling construct, including IT aspects and benefits from the simulation of many BPM IT platforms allowing the deployment and automated transformation to the execution of BPMN processes. Moreover, BPMN also enables the representation of human and technical resources, which are required in the MDSEA principles of representation. BPMN also has the advantage to provide a metamodel developed by OMG, which facilitates the implementation of the language.

9.4.5 Simulation Model Orchestration at TIM Run Time

According to the previous works published in References [35], MDSEA was already instantiated to use the simulation to support decision-making. In this chapter, the authors considered the supply chain context and looks for simulation of different simulations derived from domain specific models of Sect. 5.1.1. According to Fig. 9.9, the first step of the decision-making cycle is started by the decomposition of the decisions and the information (e.g., simulation needs and performance indicators (PIs) related to simulation objectives) supporting those decisions. This

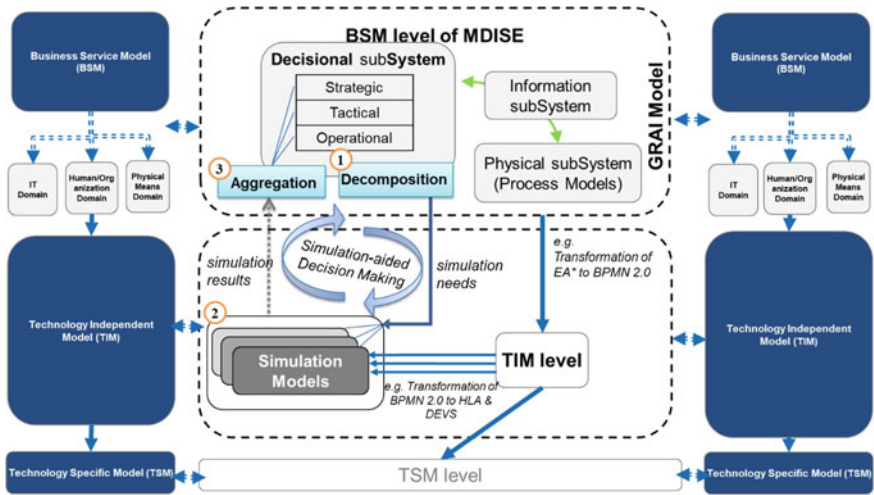


Fig. 9.9 Simulation-based decision aid for enterprise interoperability at the TIM level

step can be performed using decisional modeling methods such as GRAI grid (see Sect. 3 description). Then, several simulation solutions should be selected according to the required information treatment (see (2) in Fig. 9.9). For instance, in a manufacturing system, the decision at a strategic level can deal with the choice of handling correctly at the frontier between two enterprises the structure of data. Therefore, the simulation needs a distributed approach gluing together different simulations coming from different domain specific models. The solution intends to provide an overall mechanism to overpass interoperability barriers according to situations in the ICT supply chain domain for a given period.

As discussed in Sect. 2, the distributed simulation and HLA standard can be used to overcome these barriers. HLA is providing interoperability regarding data and time management. Then, the methodology can provide facilities the transformation of TIM workflow models into distributed discrete event models (e.g., HLA DEVS; FMI/FMU [39]) to support data exchange scenarios between domain specific simulations in order to dynamically observe the behavior of domain specific models coupled to orchestrate data exchange thanks to the run of these models. Practically, the methodology will propose a reference ICT model library to support a quick set up of the data exchange models according to the class of organizations that participate in the ICT supply chain. Of course, the library of dataflows models should include the features to integrate the main simulation flows and solutions described at the TIM level in the modeling and simulation data scenarios. The MSTB evolved will implement these models and simulation. MSTB evolved is under development but has not been released yet by the international group of researchers. This work is done in the frame of TF2 Model-Driven Interoperability, Engineering and Simulation of Interop V-Lab ([40], p. 2).

After the simulation, it is usually required to aggregate the results according to the same criteria of decomposition or enterprise layers (see (3) in Fig. 9.9). In the above example, the information about data interoperability barriers should be classified and aggregated based on annual objectives of data reliability, category of barriers faced and fixed, overall cost of data security interoperability, etc. The simulation models enabling the “simulation-aided decision-making cycle” can be the result of transforming physical subsystem models (e.g., ICT data processing models). The modeling work can be guided by ICT ecosystem techniques. The next step consists in connecting the conceptual models to the level that considers the architecture and the platform environment (e.g., IT/OT). Considering the structure of the ICT supply chain system described in Fig. 9.9, each stakeholder receives information from partners or from physical subsystems. It forms several data processes.

In the data exchanged described previously, interoperability issues can occur, where it can be vital to prepare the ICT ecosystem to react, evolve, and adapt. A simulation-aided decision cycle can be used to validate process behavior scenarios. We propose a life cycle in Fig. 9.9 to train ICT stakeholders facing different threat situations. Here, we emphasize on the importance of a modular structure, covering, and connecting different enterprises faced to interoperability barriers. The use of simulation tools is a decision aid approach to keep business continuity and business resilience. Then, in ICT systems, it is not always possible to simulate the whole data process at the operational level due to the amount of information at runtime; thus, an abstracted scenario of interoperability cartography will be run in anticipation to observe the global behavior of the system in order to prevent business continuity breaks.

9.4.6 Physical Infrastructure Interoperability with Simulation Model at TSM

The TSM level is performing a holistic and technical-oriented interoperability Analysis, on all ecosystems SC-ICTS, including data workflows model of interconnected supply chains and based on Sect. 4. To do that, this task proposes to implement the interoperability assessment method based on good practices requirements, on the compliance of all relevant regulations’ requirements like NIS, ISO 27001 series [41], including sectoral regulation such as ISO 21434 automotive regulation [42] and focusing on combined and sophisticated interoperability analysis.

This interoperability assessment methodology is adapted to such a complex ecosystem with interlaced business manufacturing processes and value networks. It includes stakeholder criticality assessment regarding their accountabilities, roles, and accesses during production, supply chain, business continuity and crisis management, and multi-tenancy management. The aim of this level will be to ensure simulation coherence regarding all interfaces and interdependencies present in this complex ecosystem at all OT and IT levels, i.e., from RTUs, PLC, DPC, SCADA, ICS, OT, to IT and cloud, and at the interfaces between IT, OT, security, and safety infrastructure in order to highlight ecosystem simulation requirements.

This level will also propose a GAP analysis and deployment plan based on models defined in the previous sections and available data. It will start from the identification of known potential technical issues in the ICT domain. Then, it will include the proposition of a list of prioritized actions according to issue categories described at the TIM level. The objective is to identify from the recent research some data interoperability threat description and to anticipate reactions with the list of actions to recover. It will prepare the action of implementation according to models of anticipated interoperability issues. If the interoperability issues were not to be fully anticipated at the modeling time, the models can describe some interoperability exception handling and management that can be developed in implementation works. In that case, the resilience, to permit to keep exchanging data in a degraded mode operation, will be described.

The simulation evaluation and TSM principle is based on the comparison between the current situation (AS-IS top BSM model coming from Sect. 4.1), the projected situation (TO-BE bottom BSM model from Sect. 4.1) and interface process model at TIM level (Sect. 5.1.3). It is prepared from the previous level to identifying more quickly and efficiently the simulation levers at TSM and actions to be carried out to eliminate this gap.

The impact assessment will lead to the choice of an appropriate action at the TSM level. The depth of analysis to be conducted will be driven by the data structure and workflow paths. Indeed, the understanding of the existing situation and especially the analysis of the gap and the levers to implement require the relevant use of methods and tools for diagnosis and problem-solving. This task will ensure to keep in sight the business level in the technical project and to prepare implementation works. The second phase will be performed to revise, upgrade, and improve the results to refine the target architecture.

9.4.7 MDISE and MSTB Evolved for CPS

Cyber-physical systems (CPS) combine digital and analog devices, interfaces, networks, computer systems, with the natural and artificial physical world. They are therefore an interesting area for experimenting with the conduct of SC-ICTS interoperability. Inherent combination of interconnected and heterogeneous behaviors of CPS falls naturally in the scope that needs interoperability in their ICT supply chains so it is a clear challenge for MDISE and MSTB evolved.

MDISE support ICT interoperability processes modeling, workloads, and performances, as presented in Fig. 9.10 that details CPS concepts within the MDISE approach. To facilitate and validate this user modeling step, the MSTB evolved tool will support the user-friendly assessment of the AS-IS as well as the TO-BE models of CPS environment models according to GRAI models in MDISE. The tool will help to populate models with different data exchange scenarios. With that objective, a user interface will allow the setting of these data. Among the MSTB improvement, MSTB evolved will propose an enhanced graphical user-friendly interface including a set of description components and annotation features to easily define

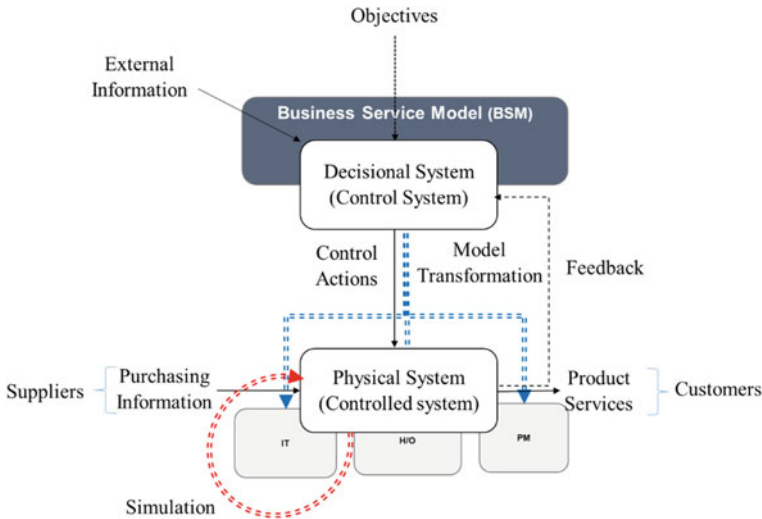


Fig. 9.10 Basic cyber-physical system (CPS) by GRAI model in MDISE

the detail of the decisional model of the control system and data workflow models of the physical model and their potential interoperability threats in the ICT system.

MSTB evolved will propose a library of model templates presenting classical activities of data exchange for CPS at the BSM top and bottom levels. For instance, the different category of connection between different types of actors will offer predefined components. Finally, MSTB evolved will embed a model transformation engine that will facilitate the creation of the reference framework of Fig. 9.10. In addition, the matching algorithms (Fig. 9.10 blue arrows) regarding MSEE version will be revisited as well as a new version of the simulation engine (Fig. 9.10 red arrows) will be proposed. The MSTB evolved will be released as an open-source tool to reach a broad adoption of a community of stakeholders. ROI for partners will be based on the customization of the tool, facilitated model building, and training of future stakeholders.

9.5 Discussion and Conclusion

Hezam Haidar, Nicolas Daclin, Gregory Zacharewicz, Guy Doumeingts.

We proposed a guideline for enterprise interoperability modeling, model transformation, and simulation. This work will help the fine models setting for the simulation and following the runs of scenarios in MSTB of potential SC-ICTS TO-BE situations to anticipate and/or correct interoperability issues. The proposition is to give more than simulation results with some aggregated information as a decision support in terms of efficiency of interoperability handling management. In detail,

the simulation will be used to run interoperability scenarios on the AS-IS models. Then, several anticipation models will be run to observe the efficiency of different interoperability plans anticipated and run to observe the gain of data interoperability. Some indicators will be implemented to observe and measure the interest of the TO-BE model proposition before going to the implementation phases at the TSM level.

The proposed extension of the model-driven interoperability is compatible with the concept of full interoperability based on federative approaches, loosely coupled organization, and reversibility concept. The main expected benefit is to remain independent of the means used within the IT, human/organizational, and physical means domains of partners. Another benefit for partners is to preserve their means by limiting the needs to strongly modify or change their means to interoperate. Conversely, the drawback is the impact of the federative approach onto the partners and the collaboration. First, from the horizontal perspective, the federative approach means to be efficient in terms of time and quality to build dynamically a model and to adapt it in case of modification from a partner. Indeed, when “something” is issued from a partner the developed interoperability must not disadvantage the collaboration in terms both of time and quality of services. Second, from the vertical perspective, the approach must be also effective to disseminate consistently the modification of higher level (BSM) toward lower levels (TIM and TSM). That means, if a new interoperability model is built, the consequences must be transmitted onto the lower models. For this purpose, mechanisms of dynamic verification and validation must be implemented at each level and between levels.

Lastly, this approach fits in with the industry 4.0 principles. Especially, it takes an interest in the cyber-physical system, data analysis and Internet of things. Indeed, the building and the adaptation on the fly of an interoperability model depend strongly on the information collected, analyzed, and treated acquired from each partner. MSTB will evolve to analyze and design of SC-ICTS and CPS. There is a challenge to keep MDISE aligned with latest development in Industry 4.0 cyber-physical system.

The chapter presents a framework and a method to guide enterprises in simulation design when they are involved in SC-ICTS. The contribution started by recalling enterprise modeling, simulation concepts, and model-driven approaches. Then, based on MDSEA, it defines the model-driven framework MDISE and methods dedicated to simulation specification and conception in B2B situation. It details the different levels required to model interoperability. It starts by top BSM models where associated recommendations to use GRAI grid models are proposed. Then, it proposes to perform a selection of domain to be represented and modeled at bottom BSM. Considering that the method needs resilience to be aware of the trends in both domains, to design a holistic and interoperable architectures based on specificity of human/organization, IT, and physical means. It drives the business requirements of the platform to the technical architecture. Then, as a transversal task, it proposes to design a conceptual high-level business-oriented interoperability simulation approach, focusing on data and services to be exchanged between domains in the platform.

In the second part, this work encourages modeling tools (such as MSTB) to evolve in order to describe dataflows models and architecture to be set up between enterprise partners at BSM and TIM. Guidelines are proposed to drive models between BSM and TIM to provide a data-driven methodology. Then, some appropriate modeling recommendations to overcome interoperability issue of CPS handling both at design and run time at TIM level are described. At the end, a holistic business-oriented interoperability modeling toolset about the B2B relation in an SC-ICTS ecosystem including potential interoperability needs has been described. It remains that metrics must be defined to attend and validate interoperability models derived along the approach.

References

1. Mourad MH, Nassehi A, Schaefer D, Newman ST (2020) Assessment of interoperability in cloud manufacturing. *Robot Comput-Integr Manuf* 61(June 2019):101832. <https://doi.org/10.1016/j.rcim.2019.101832>
2. Kozlenkova IV, Hult GTM, Lund DJ, Mena JA, Kecec P (2015) The role of marketing channels in supply chain management. *J Retail* 91(4):586–609
3. Harland CM (1996) Supply chain management, purchasing and supply management, logistics, vertical integration, materials management and supply chain dynamics. Blackwell Encyclopedic Dictionary of Operations Management. UK: Blackwell
4. Lee, E. A. (2008). Cyber physical systems: design challenges. In: Proceedings—11th IEEE Symposium on object/component/service-oriented real-time distributed computing, ISORC 2008, pp 363–369. <https://doi.org/10.1109/ISORC.2008.25>
5. Bouras A, Lagrange J-P, Rachuri S, Subrahmanian E (2007) ICT for supply chains and product lifecycle management: a research agenda for French-US collaboration. US Department of Commerce, National Institute of Standards and Technology
6. Lu T, Guo X, Xu B, Zhao L, Peng Y, Yang H (2013) Next big thing in big data: the security of the ICT supply chain. In: Proceedings—SocialCom/PASSAT/BigData/EconCom/BioMedCom 2013, pp 1066–1073. <https://doi.org/10.1109/SocialCom.2013.172>
7. Lauras M, Truptil S, Charles A, Ouzrout Y, Lamothe J (2017) Interoperability and supply chain management. In *Enterprise interoperability*. Wiley, pp 131–150
8. Vernadat F (1996) Enterprise modeling and integration. In: *Boom Koninklijke Uitgevers*. <https://doi.org/10.1177/1063293x9700500211>
9. Laamanen MT (1994) IDEF standards. *IFIP Trans a Comput Sci Technol*, pp 121–130
10. Doumeings G (1985) How to decentralize decisions through GRAI model in production management. *Comput Ind* 6(6):501–514. [https://doi.org/10.1016/0166-3615\(85\)90031-4](https://doi.org/10.1016/0166-3615(85)90031-4)
11. Le Moigne J (1977) La théorie du système général. *Théorie de La Modélisation*, PUF
12. Melese J (1972) Component analysis of systems
13. Simon HA (1960) The new science of management. Prentice-Hall, Englewood Cliffs, NJ <https://doi.org/10.1037/13978-000>
14. Mesarovic MD, Macko D, Takahara Y (1970) Theory of hierarchical, multilevel, systems. *Multilevel Syst*, 34
15. Mintzberg H (1989) The structuring of organizations. Springer, Berlin/Heidelberg, Germany, pp 322–352. <https://doi.org/10.1007/978-1-349-20317-8>
16. Simon HA (1977) The organization of complex systems. In: *Models of discovery*. Springer, pp 245–261
17. David R (1997) Modeling of hybrid systems using continuous and hybrid Petri nets. In: *Proceedings of the seventh international workshop on Petri Nets and performance models*, pp 47–58

18. Pun L (1977) Approche méthodologique de modélisation en vue de la maîtrise assistée de la production. In: Congrès de l'AFCEC Sur La Modélisation Et La Maîtrise Des Systèmes Techniques Économiques et Sociaux, 2
19. Doumeingts G (1983) Methodology to design computer integrated manufacturing and control of manufacturing unit. In: Proceedings of the international symposium on the occasion of the 25th anniversary of McGill University Centre for Intelligent Machines, Karlsruhe, Germany, pp 194–265
20. Hill AV, Giard V, Mabert VA (1989) A decision support system for determining optimal retention stocks for service parts inventories. *IIE Trans* 21(3):221–229. <https://doi.org/10.1080/07408178908966226>
21. ESPRIT Consortium AMICE (1993) CIMOSA: open system architecture for CIM. Springer, Berlin/Heidelberg, Germany, 1. <https://doi.org/10.1007/978-3-642-58064-2>
22. Mertins K, Jaekel F MO2GO: user oriented enterprise models for organisational and IT solutions. In *Handbook on architectures of information systems*, Springer, pp 649–663. https://doi.org/10.1007/3-540-26661-5_27
23. Scheer A-W (1994) Architecture of integrated information systems (ARIS). In: *Business process engineering*. Springer, pp 4–16
24. Bourey J-P, UJI RG, Doumeingts G, Berre AJ, Pantelopoulos S, Kalampoukas K (2006) Deliverable DTG2. 3 REPORT ON MODEL DRIVEN. Update 2:3–3
25. Business Process Model and Notation (BPMN) Version 2.0 (2011)
26. Lankhorst M, van Drunen H (2007) Enterprise architecture development and modelling—Combining TOGAF and ArchiMate. *Via Nova Architectura*, 21
27. Vicente M, Gama N, Mira da Silva M (2013) Using ArchiMate and TOGAF to understand the enterprise architecture and ITIL relationship. In: *Lecture Notes in Business Information Processing*, vol 148. https://doi.org/10.1007/978-3-642-38490-5_11
28. Bell M (2008) Service-oriented modeling. Wiley
29. Adamczyk BS, Szejka AL, Canciglieri O (2020) Knowledge-based expert system to support the semantic interoperability in smart manufacturing. *Comput Ind* 115. <https://doi.org/10.1016/j.compind.2019.103161>
30. IST (2008) Enterprise interoperability research roadmap (version 5.0) final. Cordis, European Commission
31. Andres B, Poler R (2020) An information management conceptual approach for the strategies alignment collaborative process. *Sustainability (Switzerland)* 12(10). <https://doi.org/10.3390/SU12103959>
32. Coutinho C, Cretan A, da Silva CF, Ghodous P, Jardim-Goncalves R (2016) Service-based negotiation for advanced collaboration in enterprise networks. *J Intell Manuf* 27(1):201–216. <https://doi.org/10.1007/s10845-013-0857-4>
33. Afsarmanesh H, Camarinha-Matos L M (2009) On the classification and management of virtual organisation breeding environments. *Int J Inf Technol Manag* 8(3):234–259. <https://doi.org/10.1504/IJITM.2009.024604>
34. Vallespir B, Chen D, Ducq Y (2005) Enterprise modelling for interoperability. In: *IFAC Proceedings Volumes (IFAC-PapersOnline)*, vol 16. <https://doi.org/10.3182/20050703-6-CZ-1902.01530>
35. Bazoun H, Zacharewicz G, Ducq Y, Boyé H (2014) SLMTToolBox: an implementation of MDSEA for servitisation and enterprise interoperability BT—Enterprise interoperability VI. In Mertins K, Bénaben F, Poler R, Bourrières J-P (eds) *Enterprise interoperability VI*; Springer: Cham, Switzerland. Springer International Publishing, pp 101–111
36. Zacharewicz G, Diallo S, Ducq Y, Agostinho C, Jardim-Goncalves R, Bazoun H, Wang Z, Doumeingts G (2017) Model-based approaches for interoperability of next generation enterprise information systems: state of the art and future challenges. *Inf Syst E-Business Manag* 15(2), 229–256. <https://doi.org/10.1007/s10257-016-0317-8>
37. Arsanjani A (2004) Service-oriented modeling and architecture. *IBM Developer Works* January, 1–15

38. Zacharewicz G, Chen D, Vallespir B (2009) Short-lived ontology approach for agent/HLA federated enterprise interoperability. In: Proceedings—2009 International conference on interoperability for enterprise software and applications, IESA 2009, pp 329–335. <https://doi.org/10.1109/I-ESA.2009.27>
39. Zacharewicz G, Frydman C, Giambiasi N (2008) G-DEVS/HLA environment for distributed simulations of workflows. *SIMULATION* 84(5):197–213. <https://doi.org/10.1177/0037549708092833>
40. TF2 model driven interoperability, engineering and simulation. INTEROP-VLab
41. ISO 27001: Risk management and compliance. 54 24 (2007) (testimony of Joel Brenner)
42. Hunjan H (2018) ISO/SAE 21434 Automotive cyber-security engineering. Presentation, Renesas Electronics LTD.
43. ICT Supply Chain (n.d.) U.S. Department of Commerce



Maturity and Accreditation

10

Tuncer Ören  and Margaret L. Loper 

Abstract

The maturity and accreditation of various educational programs are described in this chapter of the SCS M&S Body of Knowledge. It looks not only into academic education, but also into professional education.

Keywords

Modeling and simulation · Simulation education · Certified Modeling & Simulation Professional (CMSP)

10.1 Models, Programs, Processes, Individuals, and Organizations

Tuncer Ören

Grading professional maturity involves assessing processes used in a profession, individuals and organizations active in this profession, and products and/or services of the profession. Capability maturity models started in software engineering with the Carnegie Mellon capability maturity model [1, 2].

T. Ören (✉)

University of Ottawa, Ottawa, ON, Canada

e-mail: toren@uottawa.ca

M. L. Loper

Georgia Tech Research Institute, Atlanta, GA, USA

e-mail: Margaret.Loper@gtri.gatech.edu

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2023

249

T. Ören et al. (eds.), *Body of Knowledge for Modeling and Simulation*,

Simulation Foundations, Methods and Applications,

https://doi.org/10.1007/978-3-031-11085-6_10

Capability Maturity Model Integration (CMMI) is a process level improvement training and appraisal program. Administered by the **CMMI Institute**, a subsidiary of ISACA, it was developed at Carnegie Mellon University (CMU). It is required by many U.S. Government contracts, especially in software development. CMU claims CMMI can be used to guide process improvement across a project, division, or an entire organization. CMMI defines the following maturity levels for processes: Initial, Managed, Defined, Quantitatively Managed, and Optimizing. Version 2.0 was published in 2018 (Version 1.3 was published in 2010, and is the reference model for the remaining information in this wiki article). CMMI is registered in the U.S. Patent and Trademark Office by CMU.¹ [3]

Mehravari [1], (slides 42–43) clarifies when does it makes sense to use maturity models:

Requirement for a structured approach:

Demonstrated, measurable results based on an established body of knowledge*

A defined roadmap from a current state to a desired state

An ability to monitor and measure progress, particularly in the presence of change

- Response to a strategic improvement or new product/new market objective

Desire to answer these questions in a repeatable, predictable manner:

- How do I compare with my peers? (ability to benchmark)
- How can I determine how secure I am and if I am secure enough?
- How do I measure my current state? Characterize my desired state?
- What concrete actions do I need to take to improve? And in what order?
- How do I measure progress toward my desired state?
- How do I adapt to change?"

*The SCS M&S BoK Guide may be the beginning of the establishment of maturity models for M&S.

Accreditation is a management or administrative decision that decides a simulation is acceptable for particular use, a stamp of approval on the simulation from an appropriate authority. Certification and similar terms (such as confirmation) have basically the same meaning as accreditation. Sometimes the accreditation decision for a simulation involves a very formal process and is based upon V&V information developed specifically to support such a decision. In other cases, the decision is informal and may even be defacto (i.e., the simulation is simply being used for some purpose). [4]

Certification of modeling and simulation (M&S) applications poses significant technical challenges for M&S program managers, engineers, and practitioners. Certification is becoming increasingly more important as M&S applications are used more and more for military training, complex system design evaluation, M&S-based acquisition, problem solving, and critical decision making. Certification, a very complex process, involves the measurement and evaluation of hundreds of qualitative and quantitative elements, mandates subject matter expert evaluation, and requires the integration of different evaluations. [5]

¹ https://en.wikipedia.org/wiki/capability_maturity_model_integration#cite_note-1

The Certified Modeling & Simulation Professional (**CMSP**) **certification program** was created in 2002 to provide the Modeling & Simulation (M&S) industry with its own professional certification that remains valid for four years before recertification is required. The CMSP designation recognizes professionals with extensive experience and expertise in M&S. [6]

10.2 Educational Programs

Margaret L. Loper

Continuous education of Modeling and Simulation (M&S) professionals and short courses to improve the knowledge of the work force has become as important as the initial academic education, as the turnaround time for new technologies in M&S is as fast as in other high tech professions. As M&S becomes increasingly important, there is a significant and growing need to educate, train, and certify M&S practitioners, researchers, and faculty. Efforts to meet that need have taken a number of forms: academic degree programs, non-degree professional education, and professional certifications [7].

10.2.1 Academic Education

An academic degree refers to a bachelor, master's or Ph.D. program, non-degree professional education refers to skills and knowledge attained for career advancement through facilitated learning, and professional certification refers to a designation earned to assure qualification to perform a job or task.

A few universities offer M&S as an academic discipline with a degree program. Graduate-level programs are currently offered by the University of Central Florida (UCF), Old Dominion University (ODU), the University of Alabama in Huntsville (UAH), and the Naval Postgraduate School (NPS). These programs follow a traditional academic approach where students enroll in a graduate program and take a series of courses that lead to a degree. It is common for the curriculum to be based on the theory and science of the underlying subject area, and for the degree to culminate with a formal project, thesis, or dissertation.

10.2.2 Professional Education

An alternate form of M&S training is professional (or continuing) education. These courses differ from academic degrees in that they focus on applied learning rather than theory. They may also focus on a narrower area study. Professional education courses are taught by universities and industry, and come with differing levels of accreditation; thus, there are more choices in this space than for academic M&S degrees. These professional education programs serve an important role in M&S education. Many practitioners and researchers do not have the time nor desire to pursue a traditional academic degree program; however, they need to learn new skills and knowledge to be effective in their jobs.

Professional education courses typically take a few days (sometimes up to a week, or very rarely, longer) and give a broader more general topic of instruction to a student. Many M&S topics are offered by universities that teach modeling and simulation topics as graduate coursework, and it may be the same faculty members that are also teaching the short courses. However, the information in a short course is more focused, and more likely to get to the point of applicability of knowledge faster than in a graduate course that might dwell more on background and theory. Exceptions to these trends exist, however, and most institutes that offer short courses make descriptive information about the contents of the course available online.

The list of short courses below is not exhaustive in terms of the sources of such courses, nor is it exhaustive in the topics that the listed sources themselves offer. Rather, it is an example of the variety of topics in such courses, as they apply to M&S.

- Averill M. Law & Associates—seminars in topics, including discrete event simulation
- DiSTI—courses on distributed simulation technologies, graphics programming, and others
- George Mason University—modeling throughout the lifecycle, M&S basics, and others
- Georgia Institute of Technology—fundamentals and topics courses, across engineering fields
- Massachusetts Institute of Technology—topical modeling courses on traffic and other areas
- Old Dominion University—fundamentals, topics courses, and others through the VMASC center
- University of Southern California—biomedical modeling topics

10.2.3 U.S. Army M&S Professional Support

An example of a different approach to educating M&S professionals, the U.S. Army has defined two different M&S career titles to develop and train its workforce, the Functional Area (FA) 57 Simulation Operations designation for military officers and the Career Program (CP) 36 Analysis, Modeling & Simulation designation for civilians.

The FA57 is an officer with operational experience who understands military operations and training. They develop, plan, coordinate, and execute exercises at all levels of command: battalion, brigade, division, combatant command, interagency, and multi-national. FA57s are experts in modeling, simulation, and Army Battle Command Systems and facilitate the training and operational environment for commanders to conduct first-class mission planning and mission rehearsal exercises. FA57 currently has 217 authorizations in the Active Army and 209 in the Reserve Component with ranks ranging from Captain to Colonel.

The CP36 is the Department of Army's civilian Analysis, Modeling and Simulation career program, for training, educating, and developing civilian human capital in a systematic fashion. The CP36 Army Civilian Training, Education, and Development Systems was approved on 15 April 2006. Analysis, modeling, and simulation are pervasive throughout the Army and are found in the Acquisition, Analysis, Operations, Testing, Training, Experimentation, and Intelligence communities.

CP36 civilians work in a wide variety of organizations including program offices; research labs; technology, development, and engineering facilities; analysis centers; test ranges; logistics centers; headquarters; and training centers and ranges. CP36 careerists support M&S activities throughout the acquisition life cycle and in the analysis, experimentation, intelligence, operations and plans, testing, and training communities.







References

1. Mehravari N (2014) Everything you always wanted to know about maturity models. https://resources.sei.cmu.edu/asset_files/webinar/2014_018_101_293863.pdf
2. Paulk MC (2018) A history of the capability maturity model for software. Carnegie Mellon University. https://www.researchgate.net/publication/229023237_A_History_of_the_Capability_Maturity_ModelR_for_Software
3. Wikipedia—CMMI—Capability maturity model integration. https://en.wikipedia.org/wiki/Capability_Maturity_Model_Integration
4. Pace DK (2003) Verification, validation, and accreditation of simulation models. In: Obaidat MS, Papadimitriou GI (eds) Applied system simulation. Springer, Boston, MA. https://doi.org/10.1007/978-1-4419-9218-5_21
5. Balci O (2001) A methodology for certification of modeling and simulation applications. *ACM Trans Model Comput Simul* 11(4):352–377. <https://doi.org/10.1145/508366.508369>
6. NTSA-CSMP—The national training & simulation association, certified modeling & simulation professional certification program. <https://www.trainingsystems.org/cmsp>
7. Loper ML, Turnitsa CD (2017) Academic education supporting the professional landscape. In: The profession of modeling and simulation: discipline, ethics, education, vocation, societies, and economics, vol 253



Supporting Computer Domains

11

Jean François Santucci , Laurent Capocchi , Tuncer Ören ,
Saurabh Mittal , Bo Hu Li, Lin Zhang , Ting Yu Lin,
Yuanjun Laili , and Claudia Szabo

Abstract

The increase in computational capabilities affects the simulation domain. This chapter of the SCS M&S Body of Knowledge looks in the state of the art of digital

J. F. Santucci (✉) · L. Capocchi
University of Corsica, Corte, France
e-mail: santucci_j@univ-corse.fr

L. Capocchi
e-mail: capocchi@univ-corse.fr

T. Ören
University of Ottawa, Ottawa, ON, Canada
e-mail: toren@uottawa.ca

S. Mittal
The MITRE Corporation, Dayton, OH, USA
e-mail: smittal@mitre.org

B. H. Li · L. Zhang · Y. Laili
Beihang University, Beijing, China
e-mail: bohuli@moon.bjnet.edu.cn

L. Zhang
e-mail: johnlin9999@163.com

Y. Laili
e-mail: lailiyuanjun@buaa.edu.cn

T. Y. Lin
Beijing Simulation Center, Beijing, China
e-mail: lintingyu2003@foxmail.com

C. Szabo
University of Adelaide, Adelaide, Australia
e-mail: claudia.szabo@adelaide.edu.au

simulation, mobile simulation, and wearable simulation. It further describes cloud-based simulation and high-performance simulation, with additional sections on parallel evolutionary algorithms and extreme scale simulation.

Keywords

Modeling and simulation • Digital simulation • Mobile simulation • Wearable simulation • Cloud-based simulation • High-performance simulation

11.1 Digital Simulation

Jean François Santucci

Today, digital simulation is used in many areas of research and development: mechanics, fluid mechanics, materials science, astrophysics, nuclear physics, aeronautics, climatology, meteorology, theoretical physics, quantum mechanics, biology and chemistry as well as in the humanities: demography, sociology, etc. It also operates in sectors such as banking and finance. In the logic of making faster, better ... and cheaper, digital simulation has all the advantages. New challenges have appeared, of an economic nature: reactivity, anticipation and competitiveness (productivity gains). But also, safety and security issues thanks to a better understanding of accident situations in fields as varied as nuclear, automotive or aeronautics. Simulations can be used to predict failure or faults to determine the optimal time when a part should be repaired or replaced. All this concerns the products, but there are also simulations of entire factories to simulate their operations and simulations of the human body.

Numerical simulation designates the process by which a program is executed on a computer in order to represent a given phenomenon. Scientific numerical simulations are based on the implementation of theoretical models. They are therefore an adaptation to numerical means of mathematical models. They are used to study the functioning and properties of a system and to predict its evolution. A model is the translation of a phenomenon in the language of mathematical equations. The modeling of the phenomenon studied consists in taking into account fundamental principles, such as the conservation of mass and energy, and in determining the essential parameters for its description both simple and realistic. At each point of the object considered, several physical quantities (position, speed, temperature, etc.) describe its state and its evolution and allow to fully characterize its movement. These quantities are not independent but linked together by equations, which are the mathematical translation of the laws of physics governing the behavior of the object. There is only one step between the equations and the simulation codes: coding, called “discretization of equations”. This operation translates the equations into computer language, the only one understood by the computer.

There are two main approaches to numerically solve the mathematical equations of a model. The deterministic calculation method, which solves the equations after having discretized the variables, and the statistical (or probabilistic) calculation method. There are many deterministic methods: finite volumes [1, 2], finite elements [3], level set [4, 5], etc., whose use depends in part on the equations considered.

The second method, known as “Monte-Carlo” [6–10], is particularly suited to phenomena characterized by a succession of stages during which each element of the object can undergo different possible events a priori. From step to step, the progress of the sample will be determined by random draws (hence the name of the method). The deterministic and Monte-Carlo methods are the subject of numerous mathematical studies to specify their convergence in space, that is to say the variation of the precision of the approximation as a function of the number of meshes or of elements of the object, or their convergence in time, i.e., the variation of the precision according to the “time step” of calculation (delay between two instants of calculation). This calculation software or “codes” are the translation, through digital algorithms, of mathematical formulations of the physical models studied.

Another important field of digital simulation concerns Digital Testing. Digital simulators are becoming a standard and necessary CAD tool in the circuit design process. Digital simulation is the process of modeling, by computer programs, the behavior of a digital network. Digital fault simulation is the modeling of a digital network in the presence of a physical defect. The term “fault” is used to refer to some representation of a physical defect. A simple example of this is the case where an input connection to an AND gate is broken. This would be referred to as a stuck-at-1 fault [11–14].

A deductive method of fault simulation which deduces the faults detected by a test at the same time that it simulates explicitly only the good behavior of logic circuit has been developed [15]. For large logic circuits (at least several thousand gates), it is expected to be faster than “parallel” fault simulators but uses much more computer memory than do parallel simulators. The simulator is table driven, employs selective trace, models the dynamic behavior of circuits reasonably well but does not perform race analysis, and accommodates both synchronous and asynchronous circuits. It simulates three logic states: ZERO, ONE and DON’T KNOW.

The Concurrent and Comparative Simulation (CCS) which allows several simulations of a system in one single pass is also an important feature in digital simulation [16]. One of the first applications of CCS has been the Concurrent Fault Simulation (CFS) for fault simulation in digital systems described at the gate level. However, nowadays digital designers focus on more abstract languages such as VHDL (Very high-speed integrated circuits Hardware Description Language) rather than on these logical models [17–19]. Modeling and simulating of the digital circuit behaviors are possible using these languages, but they do not allow the concurrent simulation of faulty behaviors, also simply called faults. Technical barriers for the design of a concurrent fault simulator are on the one hand the lack of realistic fault

models and on the other hand the difficulty to integrate the concurrent algorithms into a simulation kernel. To reach this objective, the BFS-DEVS formalism (Behavioral Fault Simulator for Discrete Event system Specification) has been proposed [20]. This approach based on the DEVS formalism allows to model and simulate behavioral faults on discrete event system such as digital circuits described with VHDL. The BFS-DEVS simulation kernel integrates the CFS concurrent algorithms and is based on a propagated fault list technique inside the models of the system. This technique speeds up the simulation process since it allows the simultaneous detection of several faults and also simplifies result observability at the end of the simulation.

11.2 Mobile Simulation

Laurent Capocchi

Modeling and simulation (M&S) as a service (MSaaS) [21] is a concept based on the as-a-service model of cloud computing that combines web services and M&S applications. MSaaS frameworks offer effective simulation environments that can be deployed and executed on-demand by the users. They discover new opportunities for working together and enhance their operational effectiveness, saving costs and efforts in the process. In a typical MSaaS platform, user can access to M&S functionalities as services by using browser or smart client. All the M&S services are stored in cloud and are accessible using smart clients that can embed web applications. However, while the number of MSaaS tools is growing [22–26], they need to propose some important features in the dynamic simulation realization (add/remove models that change the model structure during the simulation) and the real data acquisition for simulations.

M&S of complex systems is a discipline dedicated to the field of engineering and research that tends to be exploited more and more by users and developers of mobile apps. Initially, the models and simulators were dependent on a specific application domain and they were developed by a team of engineers/researchers specialized in a given field of application (faults simulation in digital circuits, forest fires simulation, healthcare simulation, etc.). The analysis and development of a model and its simulator were therefore carried out by specialists whose working environment was confined to the workstation of a company or a research laboratory. The resulting computer programs were run locally using local input data (e.g., from a test bench). This unconnected mode of operation is no longer relevant with the advent of ubiquitous systems that need to be accessible online [27].

A set of research questions involved by the connection between simulation, cloud and smartphones can be addressed: (i) How to interface simulation software with smartphone application programming interfaces (APIs)? (ii) How to combine discrete event simulation, cloud storage (with web services interfaces) and mobile terminals? (iii) How to use a generic approach to deal with these problems?

To address the three previous research questions, a new web-based simulation approach including a collaborative discrete event M&S environment associated with a mobile app in order to remotely simulate discrete event models libraries via web services can be envisioned. A possible approach can be emerged from the domain of the design of System of Systems (SoS) [28, 29] based on discrete event simulation. The generic approach allows to automatically deploy a simulation model built by a team of engineers on a mobile terminal (smartphone, tablet, etc.). When the system is modeled and validated (by simulation) in a M&S software, it can be proposed to the end-users to remotely simulate the resulting model of the studied system with data acquired from mobile terminal sensors immersed in a real physical environment. The benefit of the approach and its motivation are highlighted through the ability (i) to perform data acquisition from mobile terminal sensors in order to simulate a system in a real physical environment and (ii) to interact with the dynamic simulation model from the mobile app during simulation.

DEVS has been introduced as an abstract formalism for the modeling of discrete event systems and allows a complete independence from the simulator using the notion of abstract simulator. This explicit separation between the modeling and the simulation allows to perform development of models without worrying about their simulation that can be invoked by web services through mobile app. DEVSimPy-mob [30, 31] gives the option of executing DEVSimPy [32] models online from mobile terminals. This approach allows an automatic cross-platform mobile app generation in the context of MSaaS services with the addition of the possibility to interact dynamically with the simulation model (by adding/removing models during simulation) and perform simulation from real data (from mobile terminal sensors).

11.3 Wearable Simulation

Tuncer Ören

Wearable simulators can be used for several reasons, such as control, training and education. For example, [33] developed a virtual reality model simulation in order to control a drone using a wearable device in a 3D environment.

Wearable simulators are used in healthcare training and education [34].

Wearable simulations are to be also useful for military training. An evaluation of wearable simulation interface for military training is done by Taylor and Barnett [35].

A special analysis of power and execution simulator for wearable devices is done by Yankov [36].

A philosophical discussion of wearable simulations is offered by Kullman [37] in “Prototyping bodies: a post-phenomenology of wearable simulations.”

11.4 Cloud-Based Modeling and Simulation

Saurabh Mittal, Bo Hu Li, Lin Zhang, Ting Yu Lin

Modeling and simulation are two distinct activities. Likewise, cloud-based modeling and cloud-based simulation require different capabilities. Fundamentally, they require that the digital infrastructure be deployed in cloud environment and is accessible through remote mechanisms. While modeling activity requires editor workbenches that may be accessible through an Internet browser, the simulation activity requires specific simulation architecture to be cloud compliant. Transitioning any existing M&S application in a cloud-based environment requires explicit M&S infrastructure engineering. Additionally, cloud-based M&S has challenges that have largely been solved in traditional simulation system engineering [38].

The cloud-based M&S foundation is built on offering modeling as a service and simulation as a service. This implies that both the model engineering and simulation engineering activities must be service-oriented. Further, these services must be deployed in a cloud environment to realize a cloud-based M&S solution. Model engineering (ME) [39] incorporates full life cycle management of the model and credibility to the model engineering process, which includes establishing the standards, theory, methods and tools for doing modeling, and the management of the model, data, knowledge, activities and organizations/people involved in the model engineering process in a collaborative manner. ME, to be cloud-deployed, requires the availability of a model repository and the execution of the ME process through a browser-based access mechanism. There exist many such tools (e.g., NoMagic Cameo, IBM Rhapsody, Eclipse IDEs, etc.) that facilitate ME in a cloud environment. Simulation engineering incorporates the execution of a model in a computational environment by a simulator and various tools and software dependencies that are required by the simulator. However, deploying a simulator in a cloud environment is not straightforward. Simulators may require high computational resources, and the high-performance computing (HPC) community has been making large computational resources available for simulation execution for quite a long time [40]. Leveraging the HPC community body of work of the past few decades and masking it behind the service interface for cloud-enabled access are indeed the easiest solution when the simulation system is purely a software system.

The newly emerging Cyber Physical Systems (CPS) and Internet of Things (IoT) ecosystem employ cloud-based systems engineering methodologies. As Mittal and Tolk [41] explore in their recent book, CPS M&S takes the form of a Live, Virtual and Constructive (LVC) system. LVC systems that incorporate simulators at varying levels of fidelity involve both hardware and software components. While one can make the constructive (i.e., software) components available in a cloud environment, bringing virtual (may include hardware) and live (hardware and hybrid) components is not practical and requires simulation engineering to rely on specific technologies, standards and methods developed by the Distributed Simulation Engineering community engaged in LVC System of Systems (SoS) engineering.

11.4.1 Community Efforts and Emerging Standards

The Simulation Interoperability Standards Organization (SISO) initiated the cloud-based modeling and simulation (CBMS) study group in 2016 [42] to identify and document the existing M&S-in-the-cloud activities, document best practices, highlight lessons learned and identify various potential standards to facilitate adoption by other practitioners. Their focus was strictly on CBMS, and application to CPS was out of scope. The group identified several focus areas or themes into which the efforts and ideas could be organized to answer important questions or to identify best practices. Potential themes included (but were not limited to):

- Developing composable services
- Service discovery
- Security
- Deployment, management and governance of services
- DEVS modeling and other alternative modeling frameworks
- Development of a Reference Architecture
- Service-oriented architectures
- Business case analysis and return on investment
- Application of the Distributed Simulation Engineering and Execution Process (DSEEP)
- The emerging role of the cloud service provider
- Impact on Validation, Verification and Accreditation (VV&A) practices
- Data services (including terrain)

The CMBS study group organized the group in four broad areas:

- *Models, simulators and data*—Analyze cloud M&S efforts from an interoperability perspective for the models, simulators and data aspects. This includes investigating semantic model interoperability, simulation architectures and handling of structured and unstructured data.
- *Architecture*—Synthesize concepts and constructs from the current M&S architectures into a coherent vision for the future optimized for modern cloud computing and big data environments. One of the goals is to enable interoperability with legacy architectures while providing an unconstrained path to the future.
- *Cloud infrastructure*—Investigate the impact of cloud computing technologies on various aspects of M&S, including system scalability, advanced visualization, scalability of data systems and high bandwidth or low latency connections to compute and memory. Also investigate ease of integration into the Internet of Things-type scenarios, which is similar to embedding M&S into live military hardware.
- *Services*—Investigate, propose and evaluate standards, agreements, architectures, implementations and cost–benefit analysis of Modeling and Simulation (M&S) as a Service (MSaaS) approaches.

The CBMS SG produced a report [43], pending approval by the SISO Standards Activity Committee. The CMBS SG also synchronized their effort with the North Atlantic Treaty Organization (NATO) Modeling and Simulation Group (MSG) effort named NATO MSG-136 [44]. NATO MSG-136 developed a Reference Architecture for MSaaS (Fig. 11.1).

The NATO MSaaS Technical Reference Architecture is described in the form of architecture building blocks and architecture patterns. An architecture building block defines a capability, capturing among other requirements any applicable M&S and data standards, and enabling technology. An architecture pattern suggests ways of combining architecture building blocks. The idea is that the Reference Architecture is not a final product but provides a structure where more content can be added over time. In principle, all capabilities for M&S in the cloud could (eventually) be captured in this Technical Reference Architecture.

The NATO MSaaS Engineering Process (EP) is a description of the process for the development and execution of simulations within an existing MSaaS implementation (or infrastructure). An existing MSaaS implementation is assumed to have M&S Enabling Services in place (such as repository services, composition

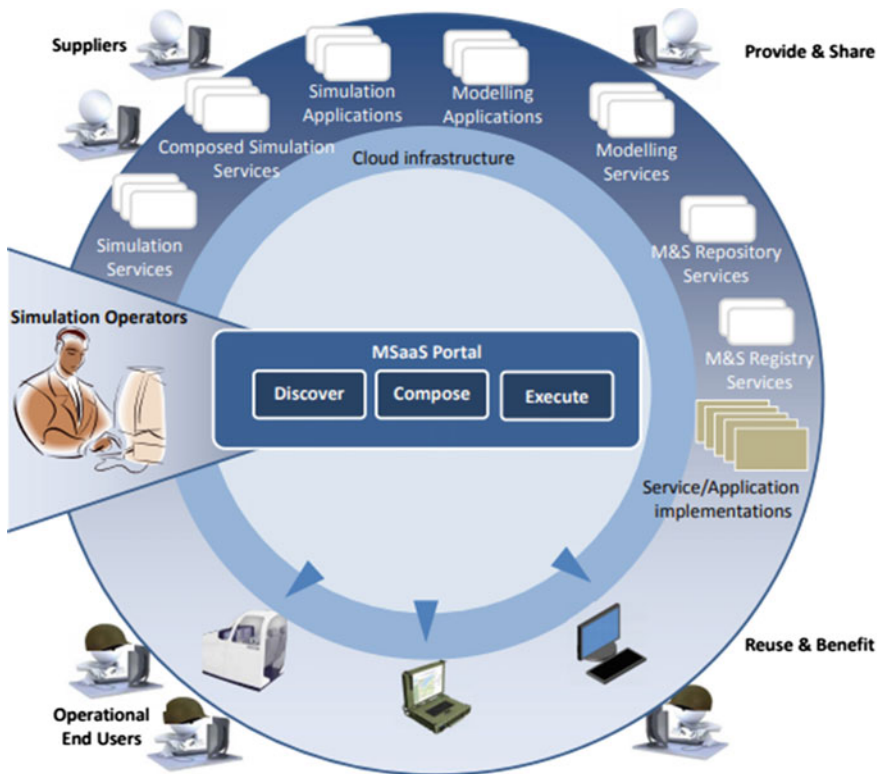


Fig. 11.1 Allied framework for MSaaS (Reproduced from [44])

services and management and control services) and provides capabilities to create a simulation. The process is described as an overlay to the DSEEP and addresses the MSaaS-specific engineering considerations during DSEEP execution. The process description will be updated as the Technical Reference Architecture evolves.

11.4.2 Theoretical Frameworks

The application of M&S employing cloud-based resources using a unified process must be fully supported by a suite of modeling languages, i.e., domain-specific languages that preserve the semantics of the specific domain. This allows engineers to decouple the domain from the model and the model from the simulation framework. This provides many benefits, since models can be constructed independently of the M&S platform, and with Domain Specific Languages (DSLs), models can still preserve the domain semantics. A modeling language also facilitates the definition of transformation from DSLs to Platform-Independent Models (PIMs).

The DEVS Unified Process (DUNIP) [45] with the underlying DEVS Modeling Language (DEVSMML) stack [46] (see Fig. 11.2) was designed to fulfill all these requirements, using as a foundation the DEVS application of discrete event dynamical system theory. They integrated Docker with the granular SOA-micro-service paradigm and advanced the state of the art in model and simulation interoperability in cloud-based M&S engineering. They described the architecture incorporating DevOps methodologies using containerization technologies to develop a cloud-based distributed simulation farm using the DEVS formalism.

Another framework called the Simulation, Experimentation, Analytics and Test (SEAT) framework by The MITRE Corporation employs the Docker technology and Continuous Integration/Continuous Delivery (CI/CD) pipelines to address integration and interoperability challenges inherent in the CPS M&S engineering

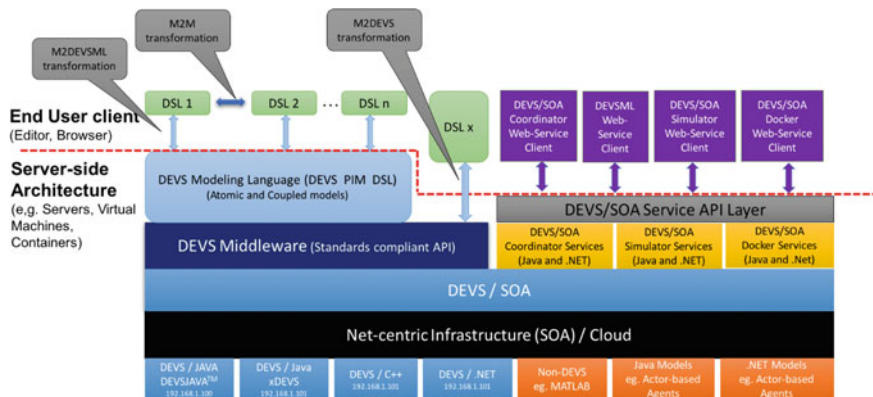


Fig. 11.2 DEVSMML stack (Reproduced from [46])

[47]. While the DEVSML stack is focused more on transforming various models into a DEVS specification, SEAT focuses on the black box approach of bringing user-apps packaged in Docker containers and providing them with a data model to interoperate with other Docker containers. The simulation environment (only the constructive in LVC) is deployed as another container app. The DEVSML stack can be subsumed in the SEAT framework as it is Docker compliant as well. Figure 11.3 shows the SEAT layered architecture framework.

Cloud simulation is a network-based (i.e., Internet, Internet of Things, telecommunications/broadcast network and mobile network) and service-oriented simulation paradigm with a new approach leading to a new ecosystem.

Utilizing cloud computing, cloud simulation can also be classified in three types of technology areas (Table 11.1):

1. Modeling theory and method
2. Simulation system theory
3. Emerging information technology usage (i.e., cloud computing, Internet of Things, service computing and high-performance computing) and specific domain application technology (Table 11.1)

The modeling theory and method relate to the MaaS and SimaaS layers in SEAT stack. The simulation system theory and emerging information technology area relates to the SimaaS, PaaS, SaaS and IaaS layers in SEAT stack. The specific domain application technology relates to MaaS, ExaaS and VisaaS layers. Table 11.1 provides further details on how various other services in these three technology areas can be conceptualized. Thus, various simulation resources and capabilities could be virtualized and encapsulated as a service to form a service pool (cloud) for coordinated and optimized management and operation. This enables the users to access on-demand simulation resources and service capabilities remotely anytime and anywhere through the terminal, network and cloud platform to support the entire simulation life cycle for system engineering and M&S-based T&E [47, 48].

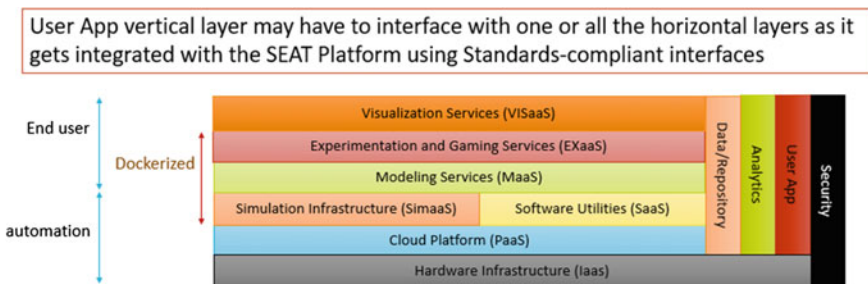


Fig. 11.3 SEAT layered architecture framework (Reproduced from [47])

Table 11.1 Technology areas

Technology area	Description
Modeling theory and method	The modeling theory and method area includes three methods: <ol style="list-style-type: none"> 1. M&S resources/capabilities/products for the entire simulation life cycle activities 2. Definition of sensing/perception/access/communication for resources/capabilities/products based on the Internet of Things 3. Description of requirements and tasks for edge/cloud simulation
Simulation system theory and emerging information technology infrastructure	The supporting environment technologies of the simulation system include: <ol style="list-style-type: none"> 1. Infrastructure-as-a-Service (IaaS) 2. Data-as-a-Service (DaaS) 3. Platform-as-a-Service (PaaS) 4. Software-as-a-Service (SaaS) 5. Simulation-Resource-as-a-Service (SRaaS) 6. Simulation-Capability-as-a-Service (SCaaS) 7. Cooperation-as-a-Service (COaaS) 8. Artificial intelligence service Simulation system construction and operation technologies include: <ol style="list-style-type: none"> 1. Virtualization-and-service-based packaging technology of edge simulation resources/capabilities/product 2. Virtualization-and-service-based packaging technology of cloud simulation resources/capabilities/products/sensing/access/communication 3. Cloud modeling/integration/simulation 4. Evaluation technology 5. Edge/cloud simulation security technology
Simulation application theory and technology	The application provides applicable terminal interaction and cloud personalized customization portal for service providers, platform operators and users. It supports four modeling simulation standards: <ol style="list-style-type: none"> 1. One agent completing one stage of simulation 2. Multiple agents cooperating to complete one stage of simulation 3. Multiple agents cooperating to complete a cross-stage simulation 4. Multiple agents obtaining simulation capability as needed

11.4.3 Essential Infrastructure

Cloud simulation supports cloud-oriented simulation task submission and edge-oriented simulation service deployment for agile systems engineering and real-time performance of simulation resources, capabilities and product integration [49]. Figure 11.4 shows a notional composition.

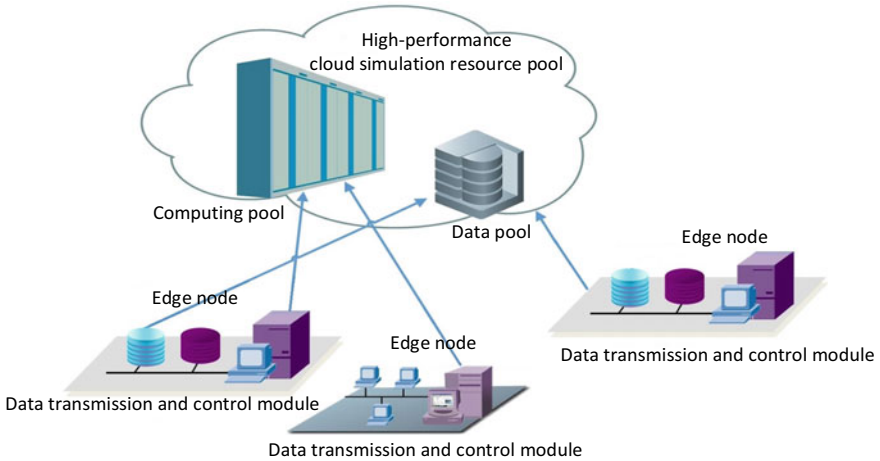


Fig. 11.4 The composition of high-performance cloud simulation

Introducing advanced computing system virtualization technology (including container-based simulation resources) and integrating other technologies for micro-service of simulation resources help achieve efficient development, composition, packaging, deployment, environment construction and operational execution [50]. Figure 11.5 shows an implementation instance of SEAT stack with more detailed service identification for some of the layers.

Semi-automated resource scheduling is adopted for container-based virtualization resources for the migration, distribution and specification of the deployment strategy of micro-service containers. Dynamic migration of simulation tasks is executed on the container core state pre-replication technology and multi-instance of micro-service during migration and runtime [51]. Figure 11.6 shows the sequence of operations for intelligent resource scheduling and migration technology.

Due to strong reliance on information and cloud computing technologies, following trends are in play:

1. *Digitization*—Combined with the Industrial Internet, cloud simulation fully supports model-based systems engineering (MBSE) and the development and integration of various simulation application APIs throughout the product life cycle. Efforts are being made to incorporate MBSE tool chains with cloud-based simulation environments for an end-to-end digital engineering ecosystem.
2. *Networking/Cloudification*—Through full cloudification, cloud simulation enables more implementation and simulator resources to be virtualized and serviced, based on the virtualization and service of various digital simulation

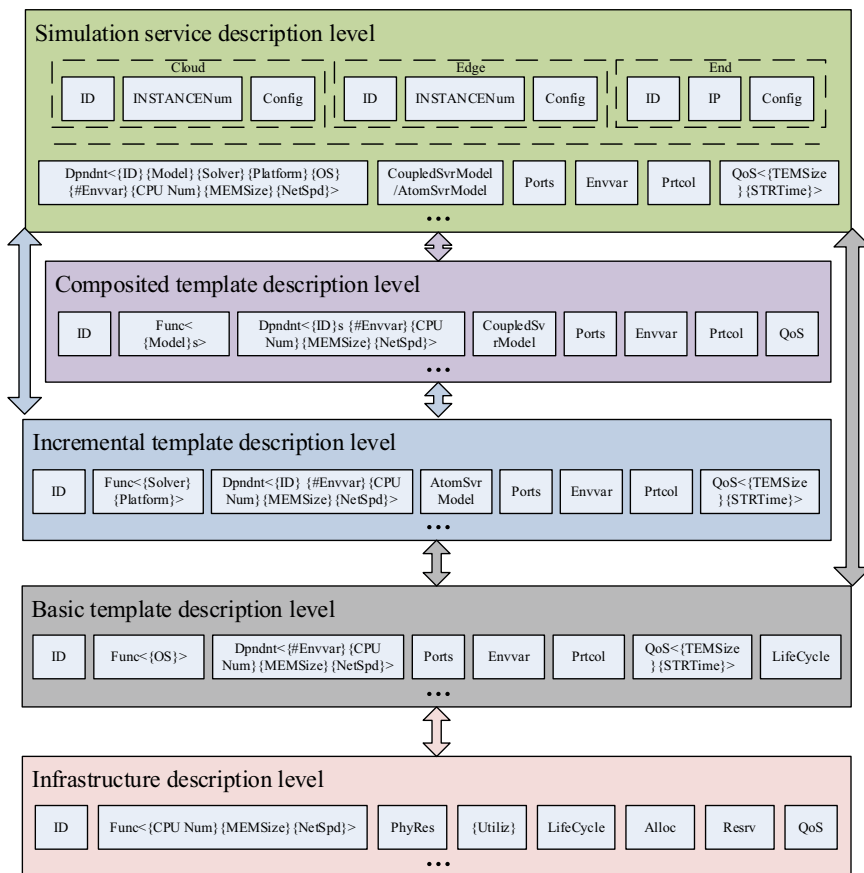


Fig. 11.5 The architecture of simulation resource virtualization and service technology

resources. The design of service layers and their integration with the backend networking and cloud platforms will continue to increase efficiency and automation.

3. *Incorporation of AI-based scheduling components*—The objective of cloud simulation is to realize a user-centered virtual simulation and experimentation environment, effectively shielding the complexity of the simulation system composition and its integration and intelligently organizing and scheduling various simulation resources/capabilities/product services. New scheduling mechanisms and intuitive user interfaces will continue to increase adoption and value addition for an integrated cloud-based modeling and simulation.

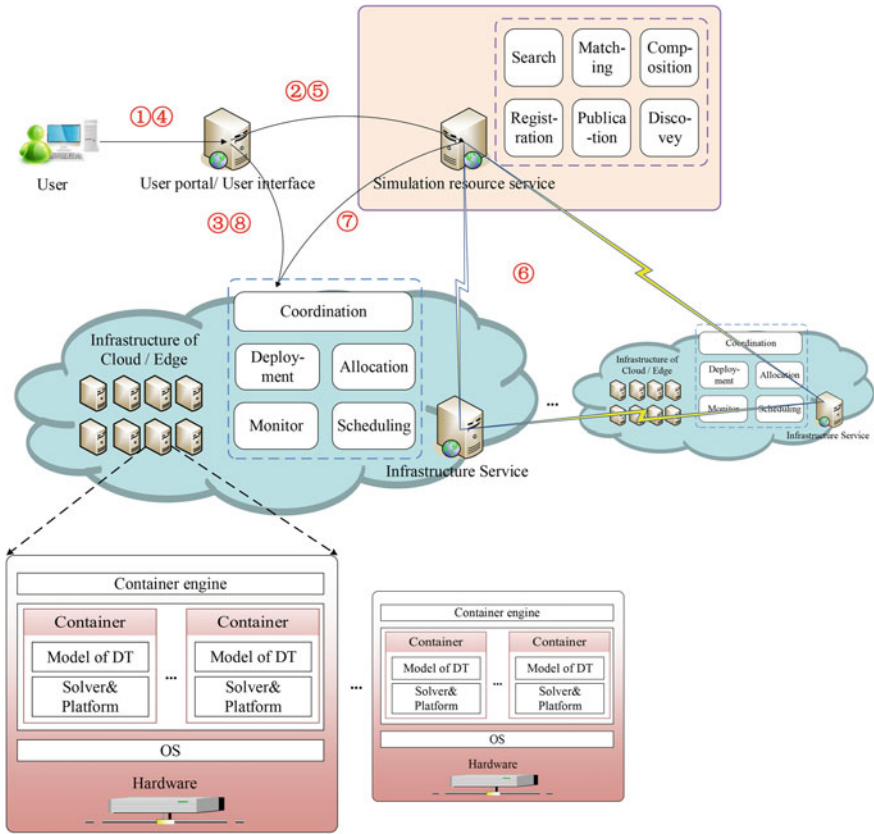


Fig. 11.6 Intelligent resource scheduling and migration technology

11.4.4 Illustrative Use Case Application

We now briefly illustrate the application of cloud simulation technology for aircraft landing gear system simulation [52]. Figure 11.7 shows the block diagram description of the system. The system involves input from multiple disciplines such as mechanical, electronics, hydraulics and control. The process of applying cloud simulation technology is as follows:

1. Package various simulation model services and software tools based on light-weight virtualization technology and configure the container in the cloud simulation environment to form a simulation service pool.
2. Users make requests through cloud modeling services on the cloud simulation portal, which can automatically combine various simulation model services and software tool services as needed to construct a simulation system in a dynamic way.

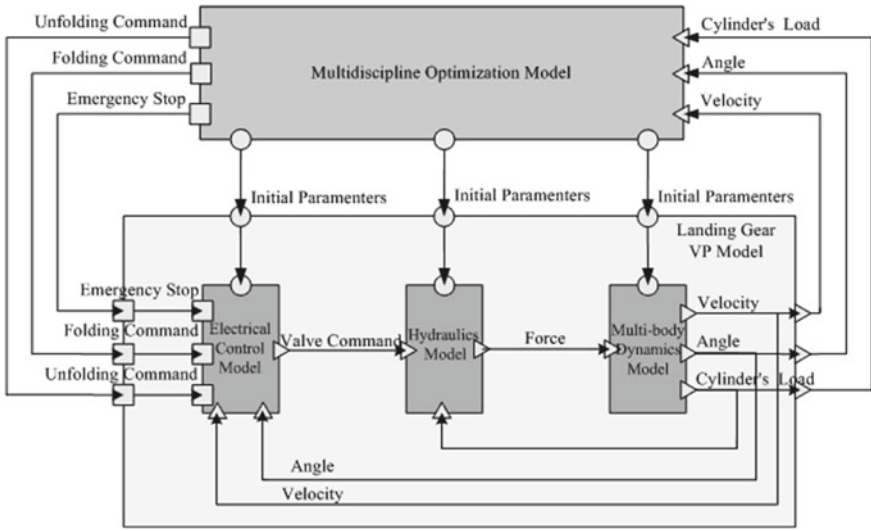


Fig. 11.7 The composition of aircraft landing gear system simulation

- 3. Users perform simulation tasks interactively on the cloud simulation portal, complete the collaborative operation of the simulation tasks which is supported by co-simulation middleware service and finally obtain satisfactory results through the portal (Fig. 11.8).

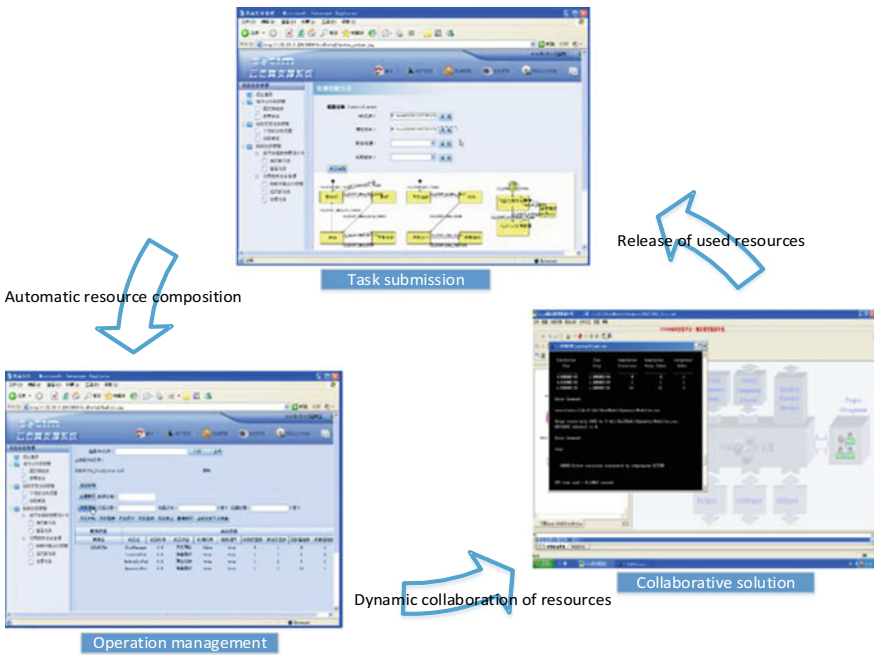


Fig. 11.8 The application process of cloud simulation

11.4.5 Synopsis

M&S methods and applications have advanced with exponentially increasing computing technology over the past few decades. Despite the amount of computing power available in a desktop/laptop today, the increased complexity of M&S applications warrants the use of HPC farms and remote computing power for offloading tasks that require high computational resources. In the past decade, the computing technology has largely moved to cloud-based environments. Consequently, M&S applications are evolving both in hardware and software to migrate to cloud-based environments. This migration is not straightforward and requires the development of various standards, best practices and sustained effort to integrate various legacy M&S applications with the latest infrastructure. This section presented the community efforts undertaken by the NATO M&S and SISO groups in laying the framework for MSaaS. We also described layered architectural approach being used in developing MSaaS architectures for integrating various DSLs with cloud-based M&S infrastructure. Finally, we demonstrated a sample workflow that illustrates the usage of a cloud-based M&S environment.

11.5 Quantum Simulation

Tuncer Ören

Some terms used in M&S have double meanings. For example, “computer simulation” means: (1) simulation of computer systems (similar to “traffic simulation” or simulation of traffic systems) and (2) simulation where all computations are done on a computer (as opposed to manual simulation). Similarly, “quantum simulation” means: (1) simulation of quantum systems and (2) simulation performed on a quantum computer (like digital, analog or hybrid simulation). The origin of quantum simulation was Feynman’s keynote at the first conference on physics and computation in 1981 at MIT [53].

11.5.1 Simulation of Quantum Systems

“Quantum simulator” is a short form to denote “simulation of a quantum system.”

This Nature Physics Insight surveys the progress made so far. The series of articles review the state of the art for quantum simulators based on atomic quantum gases, ensembles of trapped ions, photonic systems and superconducting circuits. The list is by no means exhaustive; quantum simulations are being implemented in, or have been proposed for, a number of other systems—among them nuclear spins addressed using NMR methodology, and electron spins in quantum dots or in point defects. [54]

11.5.2 Simulation Performed on Quantum Computers

“Quantum computers are machines that use the properties of quantum physics to store data and perform computations. This can be extremely advantageous for certain tasks where they could vastly outperform even our best supercomputers.” [55]. Some news about quantum computers can be found in [56]. Even though some quantum computers working only with limited number of qubits (quantum bits) currently exist [57], commercial-grade quantum computers do not yet exist; but the research and development efforts continue [58, 59] and “the quantum race is already underway” [60].

The speed supremacy of quantum computers is uncontestable:

In 2012, theoretical physicist John Preskill came up with a formulation of quantum supremacy, the superiority of quantum computers. He named it the moment when quantum computers can do things that are not possible for ordinary computers.

Seven years later, in autumn 2019, Google's quantum computer Sycamore reached this milestone. In 200 seconds, the machine performed a mathematically designed calculation so complex that it would take the world's most powerful supercomputer, IBM's Summit, 10,000 years to do it. This makes Google's quantum computer about 158 million times faster than the world's fastest supercomputer.

With the speed supremacy of quantum computers, many advanced research in many disciplines (such as science, cosmology, physics, molecular modeling, material research, chemistry, drug design and development, artificial intelligence, cybersecurity and weather forecasting) as well as their simulations will become possible with all the unique advantages of simulation.

11.6 High-Performance Simulation

Bo Hu Li, Lin Zhang, Ting Yu Lin

11.6.1 Connotation

High-performance simulation is a new simulation paradigm, new approach and new ecosystem, which combines four kinds of technologies including emerging computer science technology (i.e., cloud computing, IoT, big data, service computing and edge computing), modern modeling and simulation technology, supercomputer system technology and artificial intelligence technology, aimed to optimize the overall performance of system modeling, simulation operation and result analysis/processing for 2 kinds of users (high-end simulation user and massive simulation user group) and 3 types of simulations (constructive simulation, virtual simulation and live simulation) [61].

11.6.2 Technology System

(1) Modeling Theory and Method

High-performance parallel algorithm modeling methods mainly include four methods: job-level parallel method for large-scale simulation problems, task-level parallel method among federates of simulation system/federation, model-level parallel method among federated members and thread-level parallel method based on solving comprehensive models.

New generation digital modeling method combines typical M&S and artificial intelligent technologies [62], which is built based on correlation of data and real-time status of the physical system [63]. As high-performance simulation increasingly supports for deep reinforcement learning, intelligent algorithm modeling methods are also being applied.

(2) Simulation System Theory and Supporting Technology

The simulation system supporting environment technologies include hardware support environment technologies including technologies of heterogeneous many-core high-performance computers, big data processing components, artificial intelligence processing components, high-performance Ethernet and high-performance InfiniBand networks and software support environment technologies including operating system kernels, high-speed communication systems (MPI/shared memory/InfiniBand communication libraries) and resource management systems.

Simulation system construction and operation technologies include service technology for building and running simulation systems in high-performance environments which is implemented through virtualization services, scheduling management services, load balancing services and resource management services and enabling technologies which are related to the construction and operation of simulation systems such as multi-level parallel simulation engines, solving tools and algorithm libraries in high-performance environments.

(3) Simulation Application Theory and Technology

The simulation application theory and technology include high-performance simulation portal technologies to support high-end simulation users and mass simulation user groups for simulation task definition, submission, execution, collaboration and monitoring, high-performance technologies of virtual scene communication and virtual reality fusion and high-performance simulation application technologies related to model libraries, algorithm libraries and databases.

In addition, the new generation of digital modeling technology is used to increase the credibility of the simulation process and achieve deep integration between people, information systems and physical systems [64].

11.6.3 Key Technology

(1) High-performance Simulation-specific Acceleration Component Technology Based on Big Data and Artificial Intelligence Algorithm

Since non-mechanism modeling and simulation technologies require the support of data, algorithms and computing power, a big data and artificial intelligence processing machine based on tightly coupled high-performance computing system has been developed. The machine supports the integration of special artificial intelligence chips, and the on-demand call of real-time computing algorithm models and offline computing algorithm models, such as statistical algorithms and machine learning algorithms [61] Fig. 11.9.

(2) Interface Technology between High-performance Simulation Computer and Computer Physical System (CPS)

When CPS system accesses high-performance simulation computer system, it has the interface requirements of large amount of data, large amount of concurrency and low latency. In response to the above problem, high-performance industrial intelligent gateways are deployed at the edge of the CPS system and connected to various types of installation and simulator equipment in the simulation life cycle to support ubiquitous sensing, access and embedded simulation Fig. 11.10.

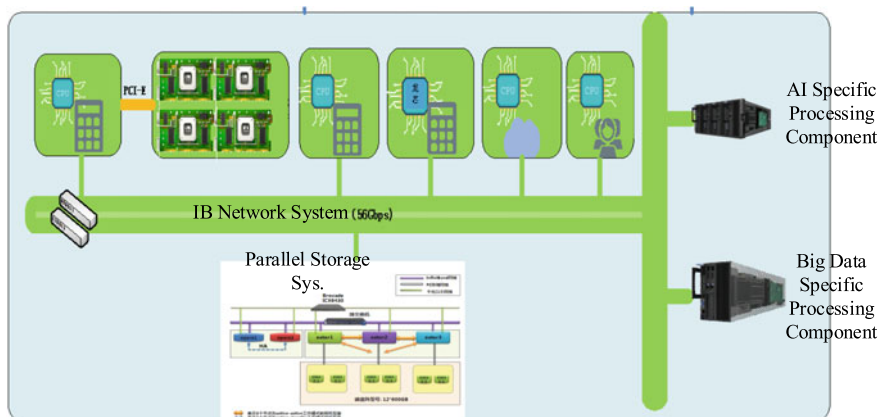
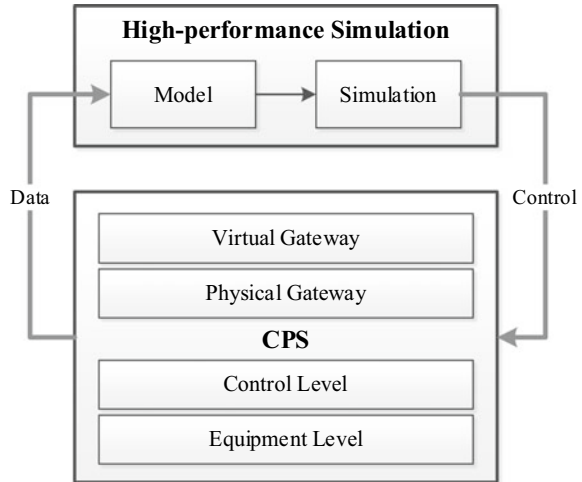


Fig. 11.9 The structure of high-performance simulation-specific acceleration component technology

Fig. 11.10 The interface technology between high-performance simulation computer and CPS



(3) Multi-level Parallel High-Performance Simulation Solving Technology

According to the characteristics of comprehensive system simulation, the requirements of fully exploiting the parallelism of the simulation system and targeted design and implementation in software are needed. An improved computer system is developed to support four kinds of parallelisms: job-level parallelism for large-scale simulation problems, task-level parallelism among members in the simulation system, model-level parallelism within federated members and thread-level parallelism based on comprehensive model solving [65], see Fig. 11.11.

(4) Parallel evolutionary algorithms

For real-world non-deterministic polynomial-time hard optimization problem, many parallel evolutionary technologies have been developed. [see Appendix: Parallel evolutionary algorithms [66]].

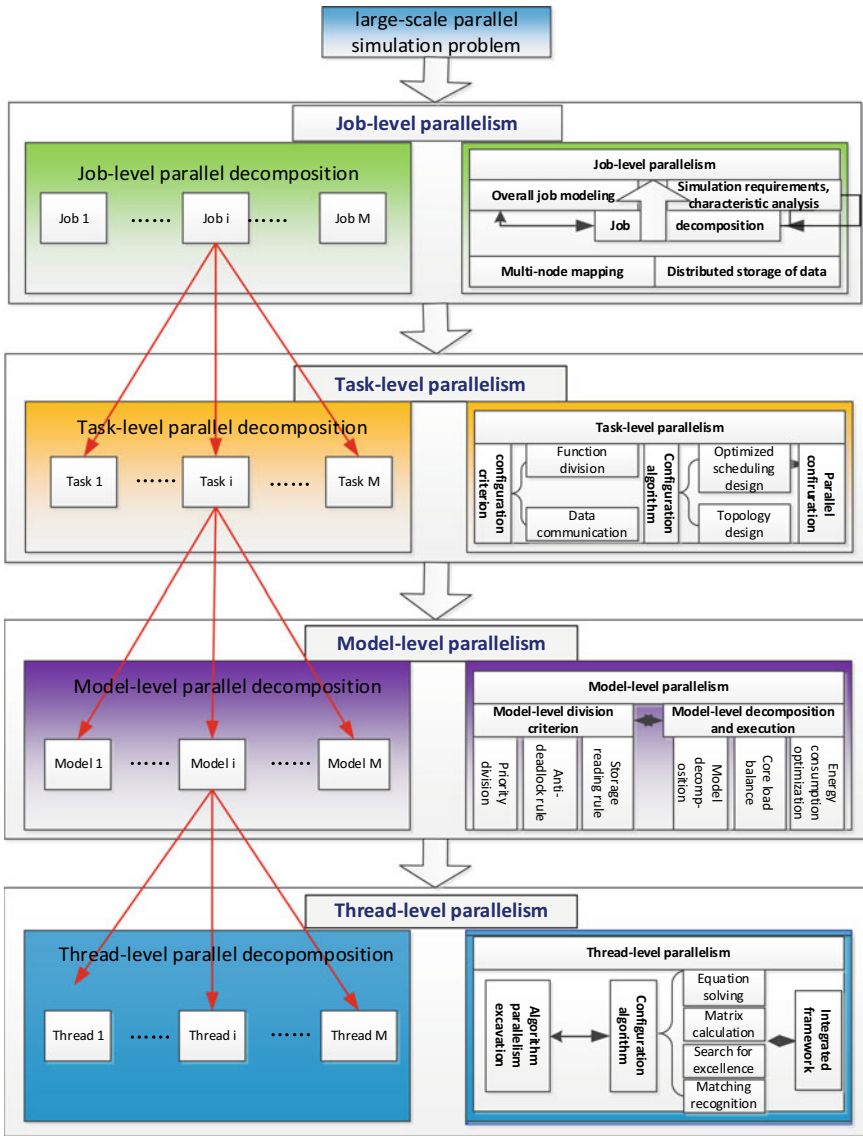


Fig. 11.11 The diagram of multi-level parallel high-performance simulation solving technology

11.6.4 Development Trend

(1) Digitizing

Combined with the Internet of Things and CPS, the simulation of digital twins is widely supported. Digital twins can evolve faster and make faster decisions.

(2) Networking/Cloudification

Networking/cloudification is based on the virtualization and servicization of various high-performance infrastructures, which supports on-demand scheduling of embedded simulation and integrated scheduling of cloud/edge simulation.

(3) Intelligentize

Intelligentize is mainly reflected in high-performance simulation languages and high-performance simulation computers, and it can effectively shield the complexity of high-performance simulation and intelligently meet the high-performance simulation needs of users in various fields.

11.6.5 Application Case

The high-performance simulation technology has been applied in research, production, operation and maintenance of comprehensive system.

As an example, a use case to show the high-performance simulation technology for aircraft control system simulation is given [65]. The system refers to multiple disciplines such as aerospace, control and power. The application process of high-performance simulation is as follows: Figs. 11.12 and 11.13.

- (1) Submit the distributed multidisciplinary virtual prototype model.
- (2) Dynamically build the operating environment of the multidisciplinary virtual prototype system.
- (3) Generate multiple instances of the multidisciplinary virtual prototype system.
- (4) Initialize the parameters of the multidisciplinary virtual prototype system (some instance).

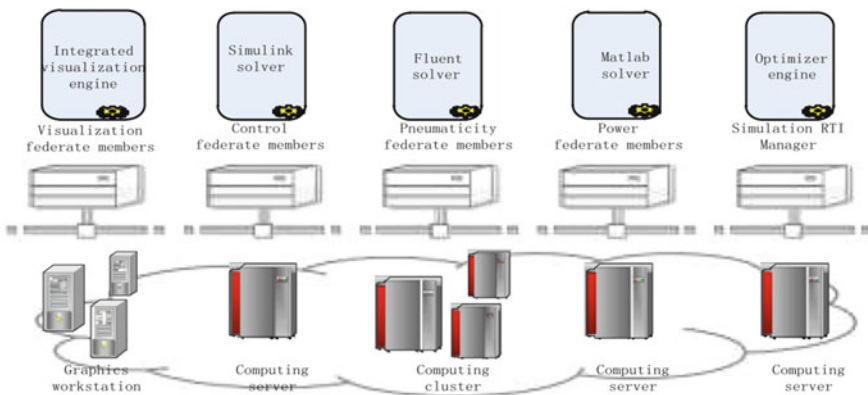


Fig. 11.12 The diagram of high-performance simulation technology for aircraft control system simulation

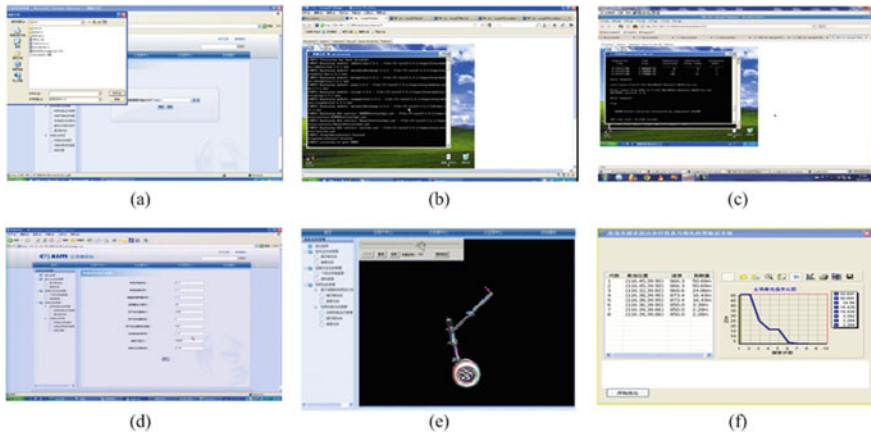


Fig. 11.13 The application process of high-performance simulation technology

- (5) Visualize the simulation process of the multidisciplinary virtual prototype system
- (6) Optimizing parallel multi-instances for the multidisciplinary virtual prototype system.

11.7 Parallel Evolutionary Algorithms

Lin Zhang, Yuanjun Laili

Real-world non-deterministic polynomial-time hard (NP-hard) optimization problems are becoming more comprehensive to solve and are presenting more challenges to evolutionary algorithms (EAs). An EA mimics natural evolution with a population in generational iterations to search for feasible and optimal solutions to NP-hard problems [67]. In dealing with these problems, parallel evolutionary algorithms (PEAs) have become increasingly popular [68]. Intuitive parallelism is to divide the population of EA into a number of sub-populations and map them onto multiple processors that work concurrently. It partitions the potential solution space, enhances global search for multi-peak problems and gives more room to maneuver for algorithm hybridization. So far, PEAs have seen many successes in solving comprehensive optimization problems [69, 70]. In recent years, three main models of PEA have been reported as design bases. These models are master–slave model, island model and diffusion model [71]. Meanwhile, hierarchical hybrid models combining one or more of the basic models have also been reported for certain cases. Owing to the widespread use of multi-core computers and clusters, island model [72–74] has become the most common, in which each sub-population evolves in an independent processor as an “island.” The “islanders” interact periodically via individual migration, in accordance with a pre-defined topology.

The resultant communication overheads are generally lower than that in master-slave model and diffusion model [75, 76]. Owing to the structure of island model, individual migration policy and island topology are the most critical elements in determining the efficiency of a PEA. Migration policy controls the migration frequency, the number of migrating individuals, the individual replacement rule and the synchronization of sub-populations [67, 77]. Much research has been reported on designing a migration policy in various scenarios, where certain offline schemes [70, 78, 79] and online strategies [80–82] are established not only to set the migration policy, but also to adaptively adjust key algorithmic parameters of sub-populations during the runtime. Island topology is also an important factor of PEA in determining the neighbors of each sub-population for individual exchanges [83]. The most commonly used ones are ring [84], mesh [85], full-mesh [86] and star topologies [87]. Generally, the island topology of a PEA is not easy to determine optimally, as communication objects of each sub-population are difficult to determine during the runtime. There are two major reasons for this. First, the correlation between the state of evolution and the topology is difficult to evaluate quantitatively. Second, the implementation means of a specific topology in a PEA are normally fixed. To deal with the above problems, studies on random topologies [88, 89] and graph-based dynamic topologies [90, 91] have been carried out. Those topologies are randomly changed during the iteration and then adapted to the problem structure [92]. However, the neighbors of each island need to be recalculated and broadcast according to the new structure in every iteration. This takes a long time, resulting in performance degradation on the parallel evolution. Today, the design of an efficient PEA with a low communication overhead remains a challenge.

One attempt to address this issue has been to tailor a PEA to the characteristics of the problem being tackled [93, 94]. Another has been to assign multiple problem-dependent heuristics to a sub-population. A third approach has been to adapt the size of sub-populations. Nevertheless, the migration policy and topology are generally kept unchanged. The deficiency of these problem-specific PEAs is that once the problem objectives or constraints change, the algorithm is hard to cope or adapt. This issue is partially addressed using memetic algorithms and hyper-heuristics with multiple EAs [95]. The most common design is to allocate a group of operators or memes (i.e., local search strategies) to different islands directly and let them interact with one another via individual migration [94, 96, 97]. However, it requires an extra algorithm selection process to update individuals inside each sub-population, which will largely degrade the parallel efficiency as well. For a multiple-EA-based PEA, as its operators and upper layer adaptation rules inside each sub-population are uniform, the search performance would be lower than those of the underlying EAs if the islands are not well balanced. Therefore, research on self-adaptation of island topology and dynamic selection of multiple EAs is imperative for PEAs.

Laili, et al. [66, 98] developed a parallel transfer evolution algorithm, which was based on the island model of parallel evolutionary algorithm and, for improving performance, transfers both the connections and the evolutionary operators from

one sub-population pair to another adaptively. Needing no extra upper selection strategy, each sub-population is able to select autonomously evolutionary operators and local search operators as subroutines according to both the sub-population's own and the connected neighbor's ranking boards. The parallel transfer evolution is tested on two typical combinatorial optimization problems in comparison with six existing ad-hoc evolutionary algorithms and is also applied to a real-world case study in comparison with five typical parallel evolutionary algorithms. The tests show that the proposed scheme and the resultant PEA offer high flexibility in dealing with a wider range of combinatorial optimization problems without algorithmic modification or redesign. Both the topological transfer and the algorithmic transfer are seen applicable not only to combinatorial optimization problems, but also to non-permuted comprehensive problems.

11.8 Extreme-Scale Simulation

Claudia Szabo.

When analyzing particle models in physics and biology where the number of particles and the complexity of their interactions scale dramatically and very quickly, extreme-scale simulation becomes a necessity. Several particle-based simulation examples, such as, among others, Amber (Salomon-Ferrer et al. [99], NAMD [100] and GROMACS [101], are used for classical molecular dynamics simulations of biomolecules. For cosmological N -body simulations, PKDGRAV [102], Gadget [103], GreeM Ishiyama et al. [104] and The Last Journey [105] are just a few examples. All these models consider complex interactions between particles whose numbers range in the trillions and thus face challenges both due to their specific parallel implementations and their extreme scale.

Mo [106] argues that parallel computing research for extreme-scale simulation exhibits three major challenges, namely limited computational scale, inadequate computing efficiency and low programming productivity. Computational scale is a challenge because the complexity of numerical algorithms increases super-linearly with the increase in the number of processor cores. Computing efficiency is a challenge due to the rapid evolution of parallel computer architectures, which affects the mapping strategies of parallel algorithms to computer architectures all the time. Lastly, low programming productivity stems from the fact that parallel programming is a very specialized skill and parallel algorithm implementations have low extensibility and portability, while at the same time physical platforms evolve rapidly. These challenges continue to widen the gap between software development productivity and supercomputer advancements [107].

Computational efficiency on modern HPC platforms is paramount. However, efficiency is challenging to achieve because few platforms have the optimal combination of memory and network (ideally fast and rich) to allow for these extreme-scale simulations to run. Iwasawa [108] gives an example of the GreeM

software to illustrate this. GreeM is designed for large-scale cosmological N -body simulations on large HPC systems, and it was used for the cosmological simulation on the K computer, which was awarded the 2012 Gordon Bell Prize. The force calculation within one MPI process is further parallelized by Barnes' vectorization algorithm, to make use of the SIMD execution units of modern computers. GreeM achieved an efficiency of more than 50%, for the cosmological N -body simulation of 102,403,102,403 particles on the entire K computer with 82,944 nodes. However, many HPC systems lack these features and thus will be unable to achieve this performance.

In this space, the Extreme-scale Simulator (xSim) [109] becomes appealing. xSim is a simulation-based toolkit for investigating the performance of parallel applications at scale. xSim scales to millions of simulated Message Passing Interface (MPI) processes. xSim employs a lightweight parallel discrete event simulation (PDES) mechanism using a conservative approach. xSim supports highly over-subscribed operations and thus has to oversee their process management and context switches, which become inherently costly.

References

1. Eymard R, Gallouët TR, Herbin R (2000) The finite volume method. In: Ciarlet PG, Lions JL (ed) Handbook of numerical analysis, vol 7, pp 713–1020
2. LeVeque R (2002) Finite volume methods for hyperbolic problems. Cambridge University Press
3. Reddy JN (2006) An introduction to the finite element method, 3rd ed. McGraw-Hill
4. Toro EF (1999) Riemann solvers and numerical methods for fluid dynamics. Springer
5. Osher S, Sethian JA (1988) Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations. *J Comput Phys* 79:12–49
6. Arsham H (1998) Techniques for Monte Carlo optimizing. *Monte Carlo Meth Appl* 4: 181–229
7. Fu MC, Hu J-Q (1997) Conditional Monte Carlo: Gradient Estimation and Optimization Applications. Kluwer, Boston
8. Liu JS (2001) Monte Carlo strategies in scientific computing. Springer, New York
9. Robert CP, Casella G (2004) Monte Carlo statistical methods, 2nd edn. Springer, New York
10. Rubinstein RY, Kroese DP (2007) Simulation and the Monte Carlo method, 2nd edn. Wiley, New York
11. Abramovici M, Breuer M, Kumar K (1977) Concurrent fault simulation and functional level modeling, pp 128–137
12. Wilcox P, Rombeek H (1976) F/LOGIC—an interactive fault and logic simulator for digital circuits, pp 68–73
13. Thompson EW, Szygenda S (1975) Digital logic simulation in a time-based, table-driven environment: part 2. *Parallel Fault Simul Computer*. 8:38–49
14. Goel P, Moorby PR (1984) Fault simulation techniques for VLSI circuits. *VLSI Design*, July 1984, pp 22–26
15. Armstrong Douglas (1972) A deductive method for simulating faults in logic circuits. *Comput IEEE Trans* C-21:464–471
16. Ulrich EG, Baker T (1973) The concurrent simulation of nearly identical digital networks. In: Proceedings of 10th design automation workshop, IEEE and ACM, New York, June 1973, pp 145–150

17. Lee J, et al (1993) Architectural level fault simulation using symbolic data. In: European design automation conference (EDAC) Feb 1993
18. Ward PC, Armstrong JR (1990) Behavioral fault simulation in VHDL. In: Proceedings of design automation conference, June 1990, pp 586–593
19. Ghosh S, Chakraborty TJ (1991) On behavior fault modeling for digital designs. *J Electron Test Theory Appl* 2:135–151
20. Capocchi L, Bernardi F, Federici D, Bisgambiglia P-A (2006) BFS-DEVS: a general devs-based formalism for behavioral fault simulation. *Simul Model Pract Theory* 14(7): 945–970
21. Procházka D, Hodický J (2017) Modelling and simulation as a service and concept development and experimentation. In: Proceedings of international conference on military technologies, May 2017, pp 721–727
22. Cayirci E (2013) Modeling and simulation as a cloud service: a survey. In: proceedings of winter simulations conference, Dec 2013, pp 389–400
23. St-Aubin B, Yammine E, Nayef M, Wainer GA (2019) Analytics and visualization of spatial models as a service. <http://cell-devs.sce.carleton.ca/publications/2019/SYNW19>
24. Zehe D, Knoll A, Cai W, Aydt H (2015) Semsim cloud service: large-scale urban systems simulation in the cloud. *Simul Model Pract Theory* 58:157–171, special issue on Cloud Simulation. <http://www.sciencedirect.com/science/article/pii/S1569190X15000805>
25. Cayirci E, Karapinar H, Ozcakir L (2017) Joint military space operations simulation as a service. In: 2017 Winter simulation conference (WSC), Dec 2017, pp 4129–4140
26. Bocciarelli P, D’Ambrogio A, Giglio A, Paglia E (2018) Model transformation services for MSaaS platforms. In: Proceedings of the model-driven approaches for simulation engineering symposium, ser. Mod4Sim ’17. San Diego, CA, USA: Society for computer simulation international, 2018, pp 12:1–12:12. <http://dl.acm.org/citation.cfm?id=3213214.3213226>
27. Barbosa JLV (2015) Ubiquitous computing: applications and research opportunities. In: Proceedings of IEEE international conference on computational intelligence and computing research, Dec 2015, pp 1–8
28. Khaitan SK, McCalley JD (2015) Design techniques and applications of cyber physical systems: a survey. *IEEE Syst J* 9(2):350–365
29. Nielsen CB, Larsen PG, Fitzgerald J, Woodcock J, Peleska J (2015) Systems of systems engineering: basic concepts, model-based techniques, and research directions. *ACM Comput Surv* 48(2):18:1–18:41, Sep 2015. <https://doi.org/10.1145/2794381>
30. Capocchi L (2019) DEVSImPy-mob. https://github.com/capocchi/DEVSImPy_mob. Accessed 10 Oct 2019
31. Kessler C, Capocchi L, Zeigler BP, Santucci J (2017) Generic architecture for interactive mobile simulation of parallel DEVS models: A missile defense application. In: Proceedings of winter simulation conference, Dec 2017, pp 1515–1526
32. Capocchi L, Santucci JF, Poggi B, Nicolai C (2011) DEVSImPy: a collaborative python software for modeling and simulation of DEVS systems. In: Proceedings of 20th IEEE international workshops on enabling technologies, June 2011, pp 170–175
33. Crespo CAB (2017) Development a virtual reality model simulation in order to control a drone using a wearable device in a 3D environment. Master’s Thesis, Instituto Politécnico de Leiria, Portugal. https://iconline.ipleiria.pt/bitstream/10400.8/3242/1/MscThesis_Christian%20Bustamante_vfinal.pdf
34. Avkin-Avwound (2021) New Avkin Avwound wearable simulator enhances wound care education. <https://www.healthysimulation.com/31550/avwound/>
35. Taylor, Barnett (2012). Evaluation of wearable simulation interface for military training. *Human factors: J Hum Factors Ergon Soc* 55(3):672–690. <https://doi.org/10.1177/0018720812466892>
36. Yankov L (2015) Master’s Thesis, KTH Royal Institute of technology, Stockholm, Sweden. <https://www.diva-portal.org/smash/get/diva2:817198/FULLTEXT01.pdf>

37. Kullman K (2016) Prototyping bodies: a post-phenomenology of wearable simulations. *Des Stud* 47, Nov:73–90. https://www.researchgate.net/publication/309298666_Prototyping_bodies_a_post-phenomenology_of_wearable_simulations
38. Sanders C (2019) Research into cloud-based simulation: a literature review (2019-SIW-031), in *Simulation Innovation Workshop, SISO, 2019*
39. Zeigler BP, Zhang L (2015) Service-oriented model engineering and simulation of system of systems engineering. In: Yilmaz L (ed) *Concepts and methodologies for modeling and simulation*. Springer
40. Obaidat MS (1993) High performance computing/computers: simulation modeling and applications. Editorial, *Transactions of SCS*
41. Mittal S, Tolk A (eds) (2019) *Complexity challenges in cyber physical systems: using modeling and simulation (M&S) to support intelligence, adaptation and autonomy*. Hoboken, NJ: John Wiley & Sons
42. SISO (2017) Cloud based modeling and simulation (CBMS) study group (SG). <https://www.sisostds.org/StandardsActivities/StudyGroups/CBMSSG.aspx>
43. Truong J, Wallace J, Mittal S, Kewley R (2019) Final report for the cloud-based modeling and simulation study group (CBMS SG) SISO-REF-nnn-DRAFT. In: *Simulation interoperability standards organization, in review*
44. Hannay JE, Berg T (2017) NATO MSG-136 reference architecture for M&S as a service. In: *Proceedings NATO modelling and simulation group symposium on M&S technologies and standards for enabling alliance interoperability and pervasive M&S applications (STO-MP-MSG-149)*. NATO science and technology organization
45. Mittal S, Risco-Martin JL (2013) Model-driven systems engineering for netcentric system of systems with DEVS Unified Process. In: *Proceedings of the 2013 winter simulation conference (WSC 2013)*, pp 1140–1151, 2013.9
46. Mittal S, Risco-Martin JL (2017) DEVSMML 3.0 Stack: rapid deployment of DEVS farm in distributed cloud environment using microservices and containers. In: *Proceedings of the 2017 spring simulation multi conference*, pp 19:1–19:12
47. Mittal S, Kasdaglis N, Harrell L, Wittman R, Gibson J, Rocca D (2020) Autonomous and composable M&S system of systems with the simulation, experimentation, analytics and testing (SEAT) framework. In: *Proceedings of spring simulation multi-conference*
48. Li BH, Chai XD, Hou BC et al (2013) Research and application on cloud simulation. In: *Summer computer simulation conference*. Society for modeling & simulation international. Toronto, Canada, 07–10 July
49. Li BH, Zhang L, Li T, Lin et al (2017) Simulation-based cyber-physical systems and internet-of-things. In: *Guide to simulation-based disciplines advancing our computational future*. Springer, pp 103–126
50. Li BH, Shi GQ, Lin TY et al (2018) Smart simulation cloud (simulation cloud 2.0)—the newly development of simulation cloud. *Asian Simulation Conference*. Kyoto, Japan: Springer, 27–29 Oct
51. Lin TY, Li BH, Yang C (2015) A multi-centric model of resource and capability management in cloud simulation. In: *Modelling & simulation*. IEEE, Cardiff, United Kingdom, 10–13 Sept, pp 555–560
52. Lin TY, Chai XD, Li BH (2012) Top-level modeling theory of multi-discipline virtual prototype. *J Syst Eng Electron* 23(3):425–437
53. Feynman R (1982) Simulating physics with computers. In: *Int J Theor Phys* 21:467–488. <https://link.springer.com/article/10.1007%2FBF02650179>
54. Trabesinger A (2012) Quantum Simulation. *Nat Phys* 8:263. <https://www.nature.com/articles/nphys2258>
55. Lu D What is a quantum computer? *New Scientist*. <https://www.newscientist.com/question/what-is-a-quantum-computer/>
56. ScienceDaily (2021) https://www.sciencedaily.com/news/computers_math/quantum_computers/

57. Discover-QC—A desktop quantum computer for just \$5,000. <https://www.discovermagazine.com/technology/a-desktop-quantum-computer-for-just-usd5-000>
58. IBM. Quantum for research. <https://www.ibm.com/quantum-computing/researchers/>
59. Siliezar J (2021) Harvard-led physicists take big step in race to quantum computing. The Harvard Gazette, July 7, 2021. <https://news.harvard.edu/gazette/story/2021/07/harvard-led-physicists-create-256-qubit-programmable-quantum-simulator/>
60. Ghose S (2020) Are you ready for the quantum computing revolution? Harvard Business review, Sept 17, 2020. <https://hbr.org/2020/09/are-you-ready-for-the-quantum-computing-revolution>
61. Li BH, Chai XD, Li T et al (2012) Research on high-efficiency simulation technology of complex system. Journal of CAEIT:221–228
62. Fujimoto R, Bock C, Chen W et al (2017) Research challenges in modeling and simulation for engineering complex systems. Springer
63. Mittal S, Diallo SY, Tolk A (2018) Emergent behavior in complex systems engineering: a modeling and simulation approach. Wiley
64. Tuegel EJ, Ingraffea AR, Eason TG et al (2011) Reengineering aircraft structural life prediction using a digital twin. Int J Aerosp Eng. Springer, pp 1687–5966
65. Li BH, Song X, Zhang L et al (2017) Cosmsol: complex system modeling, simulation and optimization language. Int J Model Simul Sci Comput:1741002
66. Laili YJ, Zhang L, Li Y (2019) Parallel transfer evolution algorithm. Appl Soft Comput J 75:686–701
67. Alba E, Tomassini M (2002) Parallelism and evolutionary algorithms. IEEE Trans Evol Comput 6(5):443–462
68. Alba E (2005) Parallel metaheuristics: a new class of algorithms. Wiley
69. Alba E (2002) Parallel evolutionary algorithms can achieve super-linear performance. Inform Process Lett 82(1):7–13
70. Jaimes AL, Coello CA (2007) MRMOGA: a new parallel multi-objective evolutionary algorithm based on the use of multiple resolutions. Concurrency Comput Pract Exp 19(4):397–441
71. Veldhuizen DAV, Zydallis JB, Lamont GB (2003) Considerations in engineering parallel multiobjective evolutionary algorithms. IEEE Trans Evol Comput 7(2):144–173
72. Cheshmehgaz HR, Desa MI, Wibowo A (2013) Effective local evolutionary searches distributed on an island model solving bi-objective. Appl Intell 38(3):331–356
73. delaOssa L, Gamez JA, Puerta JA (2006) Initial approaches to the application of islands-based parallel EDAS in continuous domains. J Parallel Distrib Comput 66(8):991–1001
74. Lin SC (1994) Coarse-grain parallel genetic algorithms: categorization and new approach. In: The 6th IEEE symposium on parallel and distributed processing, pp 28–37
75. Parsopoulos KE (2012) Parallel cooperative micro-particle swarm optimization: a master-slave model. Appl Soft Comput 12(11):3552–3579
76. Zhang XY, Zhang J, Gong YJ et al (2016) Kuhn-Munkres parallel genetic algorithm for the set cover problem and its application to large-scale wireless sensor networks. IEEE Trans Evol Comput 20(5):695–710
77. Cantú-Paz E (2001) Migration policies selection pressure and parallel evolutionary algorithms. J Heuristics 7(4):311–334
78. Lassig J, Sudholt D (2013) Design and analysis of migration in parallel evolutionary algorithm. Soft Comput 17(7):1121–1144
79. Skolicki S, Jong KD (2005) The influence of migration sizes and intervals on island models. In: Proceedings of the 2005 conference on genetic and evolutionary computation, ACM, pp 1295–1302
80. Araujo L, Merelo JJ (2011) Diversity through multiculturalism: assessing migrant choice policies in an island model. IEEE Trans Evol Comput 15(4):456–469

81. Lardeux F, Gofon A (2010) A dynamic island-based genetic algorithms framework. *Simulated evolution and learning*. Springer, Berlin, Heidelberg, pp 156–165
82. Noda E, Coelho ALV, Ricarte ILM et al (2002) Devising adaptive migration policies for cooperative distributed genetic algorithms. In: *IEEE international conference on systems, Man and Cybernetics*, vol 6, p 6
83. Matsumura T, Nakamura M, Okech J et al (1998) A parallel and distributed genetic algorithm on loosely-coupled multiprocessor system. *IEICE Trans Fundam Electron Commun Comput Sci* 81(4):540–546
84. Beckers MLM, Derks EPPA, Melssen WJ et al (1996) Using genetic algorithms for conformational analysis of biomacromolecules. *Comput Chem* 20(4):449–457
85. Fukuyama Y, Chiang HD (1996) A parallel genetic algorithm for generation expansion planning. *IEEE Trans Power Syst* 11(2):955–961
86. Yang S, Tinos R (2007) A hybrid immigrants scheme for genetic algorithms in dynamic environments. *Int J Autom Comput* 4(3):243–254
87. Miyagi H, Tengan T, Mohammed S et al (2010) Migration effects on tree topology of parallel evolutionary computation. In: *IEEE region 10 conference on TENCN*, pp 1601–1606
88. Defersha FM, Chen M (2008) A parallel genetic algorithm for dynamic cell formation in cellular manufacturing systems. *Int J Prod Res* 46(22):6389–6413
89. Defersha FM, Chen M (2010) A parallel genetic algorithm for a flexible job-shop scheduling problem with sequence dependent setups. *Int J Adv Manuf Technol* 49(1–4):263–279
90. Li L, Garibaldi JM, Krasnogor N (2009) Automated self-assembly programming paradigm: the impact of network topology. *Int J Intell Syst* 24(7):793–817
91. Whitacre JM, Sarker RA, Pham QT (2008) The self-organization of interaction networks for nature-inspired optimization. *IEEE Trans Evol Comput* 12(2):220–230
92. Amaldo I, Contreras I, Millán-Ruiz D et al (2013) Matching island topologies to problem structure in parallel evolutionary algorithms. *Soft Comput* 17(7):1209–1225
93. Jin J, Crainic TG, Lketangen AA (2014) A cooperative parallel metaheuristic for the capacitated vehicle routing problem. *Comput Oper Res* 44:33–41
94. Segura C, Segredo E, Leon C (2013) Scalability and robustness of parallel hyperheuristics applied to a multiobjective frequency assignment problem. *Soft Comput* 17:1077–1093
95. Talbi EG (2002) A taxonomy of hybrid metaheuristics. *J Heuristics* 8:541–564
96. Deng W, Chen R, Gao J et al (2012) A novel parallel hybrid intelligence optimization algorithm for a function approximation problem. *Comput Math Appl* 63(1):325–336
97. Tang J, Lim MH, Ong YS (2007) Diversity-adaptive parallel memetic algorithm for solving large scale combinatorial optimization problems. *Soft Comput* 11:873–888
98. Laili YJ, Tao F, Zhang L (2016) Multi operators-based partial connected parallel evolutionary algorithm. In: *Evolutionary computation. CEC, 2016 IEEE Congress on, IEEE*, pp 4289–4296
99. Salomon-Ferrer R, Case DA, Walker RC (2013) An overview of the Amber biomolecular simulation package. *Wiley Interdisc Rev Comput Mol Sci* 3(2):198–210
100. Phillips JC, Braun R, Wang W, Gumbart J, Tajkhorshid E, Villa E, ..., Schulten K (2005) Scalable molecular dynamics with NAMD. *J Comput Chem* 26(16):1781–1802
101. Pronk S, Páll S, Schulz R, Larsson P, Bjelkmar P, Apostolov R, ..., Hess B (2013) GROMACS 4.5: a high-throughput and highly parallel open source molecular simulation toolkit. *Bioinformatics* 29(7):845–854
102. Potter D, Stadel J, Teyssier R (2017) PKDGRAV3: beyond trillion particle cosmological simulations for the next era of galaxy surveys. *Comput Astrophys Cosmol* 4(1):2
103. Springel V, Yoshida N, White SD (2001) GADGET: a code for collisionless and gasdynamical cosmological simulations. *New Astron* 6(2):79–117
104. Ishiyama T, Nitadori K, Makino J (2012) 4.45 Pflops astrophysical N-body simulation on K computer—The gravitational trillion-body problem. In *SC'12: Proceedings of the international conference on high performance computing, networking, storage and analysis. IEEE*, pp 1–10

105. Heitmann K, Frontiere N, Rangel E, Larsen P, Pope A, Sultan I, Uram T, Habib S, Finkel H, Korytov D, Kovacs E (2020) The last journey. I. an extreme-scale simulation on the Mira supercomputer. arXiv preprint arXiv:2006.01697
106. Mo ZY (2018) Extreme-scale parallel computing: bottlenecks and strategies. *Front Inf Technol Electron Eng* 19(10):1251–1260
107. Johansen H, McInnes LC, Bernholdt D, Carver J, Heroux M, Hornung R, ..., Ndousse-Fetter T (2014) Software productivity for extreme-scale science (2014). In: Report on DOE workshop, Jan, pp 13–14
108. Iwasawa M, Namekata D, Nomura K, Tsubouchi M, Makino J (2020) Extreme-scale particle-based simulations on advanced HPC platforms. *CCF Trans High Perform Comput*:1–13
109. Böhm S, Engelmann C (2011) xSim: The extreme-scale simulator. In: 2011 international conference on high performance computing & simulation. IEEE, pp 280–286



Synergies of Soft Computing and M&S

12

Jean François Santucci , Laurent Capocchi , Tuncer Ören ,
Claudia Szabo, and Valdemar Vicente Graciano Neto 

Abstract

This chapter of the SCS M&S Body of Knowledge starts with section on fuzzy logic and simulation, neural networks, and artificial intelligence. It then provides a detailed look at the agent metaphor and agent-based simulation.

Keywords

Modeling and simulation · Fuzzy logic · Fuzzy simulation · Neural networks ·
Neural network simulation · Artificial intelligence · Agent-based modeling

J. F. Santucci (✉) · L. Capocchi
University of Corsica, Corte, France
e-mail: santucci_j@univ-corse.fr

L. Capocchi
e-mail: capocchi@univ-corse.fr

T. Ören
University of Ottawa, Ottawa, ON, Canada
e-mail: toren@uottawa.ca

C. Szabo
University of Adelaide, Adelaide, SA, Australia
e-mail: claudia.szabo@adelaide.edu.au

V. V. Graciano Neto
Federal University of Goiás, Goiânia, Brazil
e-mail: valdemarneto@ufg.br

12.1 Fuzzy Logic and Fuzzy Simulation

Jean François Santucci

In the recent years, a set of work has concerned the introduction of fuzzy notions [1–5] into DEVS concepts in order to propose incorporation of uncertainty into models [6–11]. These approaches developed around the DEVS formalism have been proposed in order to apply fuzzy set theory to the sets and functions defined on crisp sets in DEVS formalism.

Fuzzy modeling and simulation can be useful in the study of different kinds of systems for example in the case of the modeling of systems for which we have few observations of how and where the human being acts as a sensor or expert.

The fuzzy theories are a set of theories of mathematical concepts generally proved and tested, a formal framework for modeling and interpretation of fuzzy proposals (knowledge) and imperfect data. A proposal as “tomorrow there will be a lot of wind” is both inaccurate or uncertain and incomplete (according to Zadeh [1]):

- inaccurate because we cannot know how to quantify “a lot of wind”? An inaccuracy is a difficulty in articulating a fact, that is on its content “about 20 km/h”; it does not give an accurate value but an interval. It generally occurs when the data is expressed in a linguistic way as “a lot”;
- uncertain, because we do not know how to be sure that tomorrow there will be really much wind, uncertainty is a doubt about the validity of an act. It refers to the veracity of the information. It is a coefficient given to a proposal which can be true or false;
- incomplete, because from this proposal we do not know exactly the true speed of the wind at a given time. Incompleteness is a lack of knowledge, or the partial knowledge on some features of the system. They may be due to the inability to get information or to a problem which occurred during the acquiring of the knowledge. The incompleteness can be seen as a particular case of inaccuracy.

Fuzzy logic is an extension of classical logic. It was presented by Zadeh [1] as a framework for the approximate reasoning, a mathematical theory whose purpose of study is fuzzy systems. The approximate reasoning and treatment of inaccurate and uncertain facts are quite natural for human beings. For reasoning about such knowledge, classic modeling is not sufficient; in effect in this case, the approximations on variables generate, at the end, relatively large errors.

Fuzzy modeling deals with the fuzzy values from the beginning, allowing the final to obtain a range of values (inaccuracy) larger but fairer. According to Zadeh [2], fuzzy modeling provides approximate but effective ways to describe the behavior of systems that are too complex or too badly defined to allow the use of a precise mathematical analysis. The study of these systems requires consideration of inaccuracies and uncertainties but also by a relevant and efficient reasoning on the system as a whole (input and output variables, behavior).

Fuzzy set theory is a mathematical theory in the field of abstract algebra. It was introduced by Zadeh in 1965 [1], and we can infer from such a theory a new logic which bypasses the principle of excluded elements, unlike conventional membership

notions. Fuzzy logic [5] is based on the concept of fuzzy sets. The definition of a fuzzy set answers to the need for representation of inaccurate or uncertain knowledge or because they are expressed in natural language by an observer who gives little detail or is unreliable, either because they were obtained with observation instruments that produce errors or are unclear. In a reference set E , a fuzzy set f in E is characterized by a membership function μ_f which associates every element $x \in E$, the degree $\mu_f(x)$ between 0 and 1, where x is in f . The concept of fuzzy set aims to make gradations in the membership of an element x to a class f , i.e., to allow an element to belong more or less strongly to this class. The fuzzy set theory also provided a whole set of mathematical methods to manipulate the fuzzy sets.

The approaches integrating fuzzy sets into DEVS bring nice formalisms allowing to deal with fuzzy events, fuzzy transition functions, and even fuzzy time advance function. Fuzzy technology has the advantage of using explicit knowledge (the system works by applying rules) by highlighting three distinct steps: (i) Initially, the fuzzification process is studied in order to consider the inclusion of linguistic information. (ii) The second part is devoted to the identification of fuzzy rules mainly when observations on the behavior of the system are imprecise and uncertain. Fuzzy rules (such as “if X ... then Y ”) allow to express the knowledge concerning the problem to address. (iii) The last part concerns the defuzzification which is to convert the fuzzy domain to the digital domain, with a conversion preferences. The method of defuzzification is used to exploit the information encoded in the output fuzzy sets corresponding to expert knowledge on the output variable of the model.

The integration of fuzzy logic into the DEVS formalism involves three steps: (i) the definition of concepts allowing to deal with fuzzy logic in the framework of DEVS; (ii) the selection of a fuzzy logic programming language in order to accomplish the integration of DEVS and fuzzy logic; and (iii) the implementation of the concepts into the DEVSImPy framework [6, 7]. The first step allows to define the concepts allowing to perform with the DEVS formalism the three required steps involved in fuzzy logic control: fuzzification, the fuzzy rule definition and firing, and finally defuzzification. A set of available fuzzy theory programming languages (FCL—Fuzzy Control Language [12], FPL—Fuzzy Programming Language [13], FTL—Fuzzy Technology Language [14], FSTDS—fuzzy STDS [15], FSML—Fuzzy System Modeling Language [16], etc.) can be used in order to be coupled with the DEVS formalism.

12.2 Neural Networks and Neural Network Simulation

Laurent Capocchi

The artificial intelligence domain grows every day with new algorithms and new architectures. ANNs became a very interesting domain since the eighties when the back-propagation learning algorithm and the feed-forward architecture were first introduced [17, 18]. As time passed, ANNs were able to solve nonlinear problems and were being used in classification, prediction, and representation of complex systems.

Nowadays, ANNs are still in the research domain to enhance the performance and to facilitate the configuration parameters needed for the learning process [19–22]. Comparing multiple configuration parameters and learning algorithms is very common for ANN users and algorithm developers. To help users and developers test and compare different algorithm implementations and parameter configurations, the use of Comparative and Concurrent Simulation (CCS) [23] is a practical solution.

Artificial neural network since appearance is a black box capable of resolving problems that linear computer cannot. Therefore, the configuration of this revolutionary computing remains a hard task for modeler since it depends on the application complexity. DEVS formalism [24] has its hierarchical and modular aspect. This formalism gives the opportunity to defragment a system or a model in an easy way allowing interaction with the architecture or the behavior. The nature of the feed-forward ANN architecture is based on discrete messaging between neural layers or individual neurons. In this way, modeling ANN in a discrete event formalism might be considered.

DEVS is a formalism and a very interesting platform which can be adapted for multiple applications by adding extensions to it. At the same time, CCS is a technique to compare between simulations with different paths and data values [23, 25]. One of the first applications of the CCS is concurrent fault simulation (CFS) to simulate faults (mainly for digital systems) [26]. CCS can be applied to many fields, and it matches the idea of comparing multiple ANN configurations during the learning phase. This idea enables the simulation of multiple ANN configurations and compares their performances.

12.3 Synergies of Artificial Intelligence and M&S

Tuncer Ören

Gardner's theory of multiple intelligences posits that humans might have eight types of intelligences which are as follows: Linguistic, Logical/Mathematical, Spatial, Bodily-Kinesthetic, Musical, Interpersonal, Intrapersonal, and Naturalist Intelligences [27].

Cognitive abilities include:

perception,

focusing, filtering,

intention detection,

reasoning,

generalization, abstraction,

learning,

understanding,

formulating and solving problems,

asking and answering questions,

decision making,

natural language understanding and conversing in a natural language, etc. [28].

A definition of artificial intelligence follows: “*Artificial intelligence is the study of mental faculties through the use of computational models*” [29]. Another definition of artificial intelligence is as follows: “*Artificial Intelligence (AI) is a broadly defined term that encompasses multidisciplinary research and development activities involved with emulating human intelligence on machines.*” [30].

Some aspects of human intelligence are often not discussed in artificial intelligence. In fact, some humans lie and deliberately other as well as nature. These human cognitive features should not be emulated by AI. (Please see Chap. 8 on Ethics).

A list of my 45 publications and 35 presentations and other activities on the synergies of artificial intelligence, cybernetics, and simulation during 1970s through 2000s can be accessed at (Ören–AISim [31]). Starting 1990s, I switched to agent-directed simulation (ADS). A list of 88 publications and 68 presentations and other activities on software agents and ADS can be found at (Ören-ADS [32]).

For a long time, simulation contributed to the field of artificial intelligence by making possible the cognitive simulation studies. Now, simulation can benefit from the advances in artificial intelligence as well as from advances in general system theories, software and computer engineering, and mathematical modelling and experimentation techniques.

The question is not whether or not to have artificial intelligence in simulation, but rather how to have it? at which level? how reliably? how soon? and above all how intelligently?

In 1985, already over half a dozen simulation meetings are announced where artificial intelligence aspect is primordial.

The quotation on the cover of this issue would well summarize an attitude: ‘... unintelligent computerization is not enough’. [33]

For artificial intelligence systems, intelligence is a goal-directed and often adaptive knowledge processing ability.

12.3.1 Contribution of M&S to Artificial Intelligence

The term Artificial Intelligence was coined by John McCarthy in 1956. ... Two different views form the roots of AI: understanding human intelligence and building machines that surpass human intelligence.

In order to understand human intelligence, Newell and Simon [34] were the early scientists interested in the simulation of human thought processes (cognitive abilities). Simon also states a fundamental role of simulation: “The real power of the simulation technique is that it provides not only a means for stating a theory but also a very sharp criterion for testing whether the statement is adequate.” [34]

The opposite view was to build super intelligent computers having cognitive abilities far superior than the humans. ... This dichotomy in AI still continues. See for example, “Two Approaches to Machine Intelligence” . edited by Williams [35]

The field of knowledge-based systems is a branch of AI. “Three fundamental concepts of knowledge-based systems distinguish them from conventional algorithmic programs and from general search-based programs: (1) The separation of the knowledge from how it is used. (2) The use of highly specific domain knowledge. (3) The heuristic rather than algorithmic nature of the knowledge employed.” [36 Most commonly used knowledge-based systems are rule-based systems. [37]

12.3.2 Contribution of Artificial Intelligence to M&S

Cognizant simulation (or AI-based simulation) is the use of artificial intelligence in a simulation study, for the generation of model behavior and for the associated learning abilities. Table 12.1 gives a synopsis of four major categories of possibilities. ...

In a cognizant simulation environment (or AI- supported simulation), artificial intelligence is used for computer assistance for interfaces, for processing elements of a simulation study, and for their cognitive abilities such as learning abilities.

A simulation environment can have two types of interfaces: front-end and back-end interfaces. Front- end interfaces are used to specify, edit, or generate elements of a simulation problem. Back-end interfaces are used by systems to communicate to users the primary and auxiliary outputs of the system.

Table 12.2 summarizes the functions offered and which might be offered within cognizant simulation environments. ...

Table 12.1 Categories of cognizant simulation (AI-based simulation)^a

No nesting with a non-simulation knowledge-based system	(Single paradigm) simulation system	
	(Multiparadigm) simulation system	
Nested with a non-simulation knowledge-based system	(Single or multiparadigm) simulation system with a nested knowledge-based system	
	Knowledge-based system with a nested (single or multiparadigm) simulation system (with an optional <i>nested</i> knowledge-based system)	

^a AI is primarily used for the generation of model behavior (as it is the case of knowledge-based simulation and qualitative simulation) as well as associated machine learning

Table 12.2 Cognizant simulation environments (AI-supported simulation)

For elements of simulation studies	Cognitive abilities for front-end interface		For processing
	For generation	For specification/editing	
Goal	<ul style="list-style-type: none"> • Goal generation • Hypothesis formulation 	<ul style="list-style-type: none"> • Goal specification/editing 	<ul style="list-style-type: none"> • Goal processing • Goal seeking • Goal evaluation
Parametric model	<ul style="list-style-type: none"> • Modeling <ul style="list-style-type: none"> – Model formulation – Model synthesis 	<ul style="list-style-type: none"> • Model editing • Knowledge-based model editing 	<ul style="list-style-type: none"> • Model-base management • Model analysis <ul style="list-style-type: none"> – model characterization – model evaluation • model transformation
Model parameters	<ul style="list-style-type: none"> • Parameter <ul style="list-style-type: none"> – Estimation – Calibration 	<ul style="list-style-type: none"> • Knowledge-based editing <ul style="list-style-type: none"> – Parameters – Auxiliary parameters 	<ul style="list-style-type: none"> • Symbolic processing of parameters <ul style="list-style-type: none"> – auxiliary parameters
Design of experiments	<ul style="list-style-type: none"> • Knowledge-based design of experiments 	<ul style="list-style-type: none"> • Specification/editing of design of experiments 	<ul style="list-style-type: none"> • Processing (evaluation, selection, ...) of design of experiments
For every experiment; experimental conditions	<ul style="list-style-type: none"> • Knowledge-based generation of experimental conditions (represented as <ul style="list-style-type: none"> – dynamic templates or – semantic nets) 	<ul style="list-style-type: none"> • Specification/editing of experimental conditions <ul style="list-style-type: none"> – initial conditions of state variables – behavior generator – behavior generation parameters 	<ul style="list-style-type: none"> • Automatic selection of <ul style="list-style-type: none"> – behavior generator – behavior generation parameters – reliability
Simulation program	<ul style="list-style-type: none"> • Transformation of problem specifications into a simulation program • Automatic compiling/link editing necessary modules 	<ul style="list-style-type: none"> • Automated editing of simulation programs 	<ul style="list-style-type: none"> • Processing simulation programs (legacy programs, new programs) <ul style="list-style-type: none"> – program understanding – program reliability
Behavior and recommendation	<ul style="list-style-type: none"> • Monitoring behavior generation conditions • Comparing real system behavior and simulated behavior to detect possible anomalies in a real system 	<ul style="list-style-type: none"> • Automatic editing of experimental conditions 	<ul style="list-style-type: none"> • Behavior monitoring • Behavior analysis • Evaluation of <ul style="list-style-type: none"> – behavior – performance measure – decision variables
			Back-end interface

Figure 12.1 depicts elements of cognizant simulation environments. They consists of cognizant front- and back-end interfaces, specifications of single or multiparadigm (cognizant) simulation systems, knowledge-based systems used for non-experiential purposes, knowledge bases and their associated machine learning modules." [28]

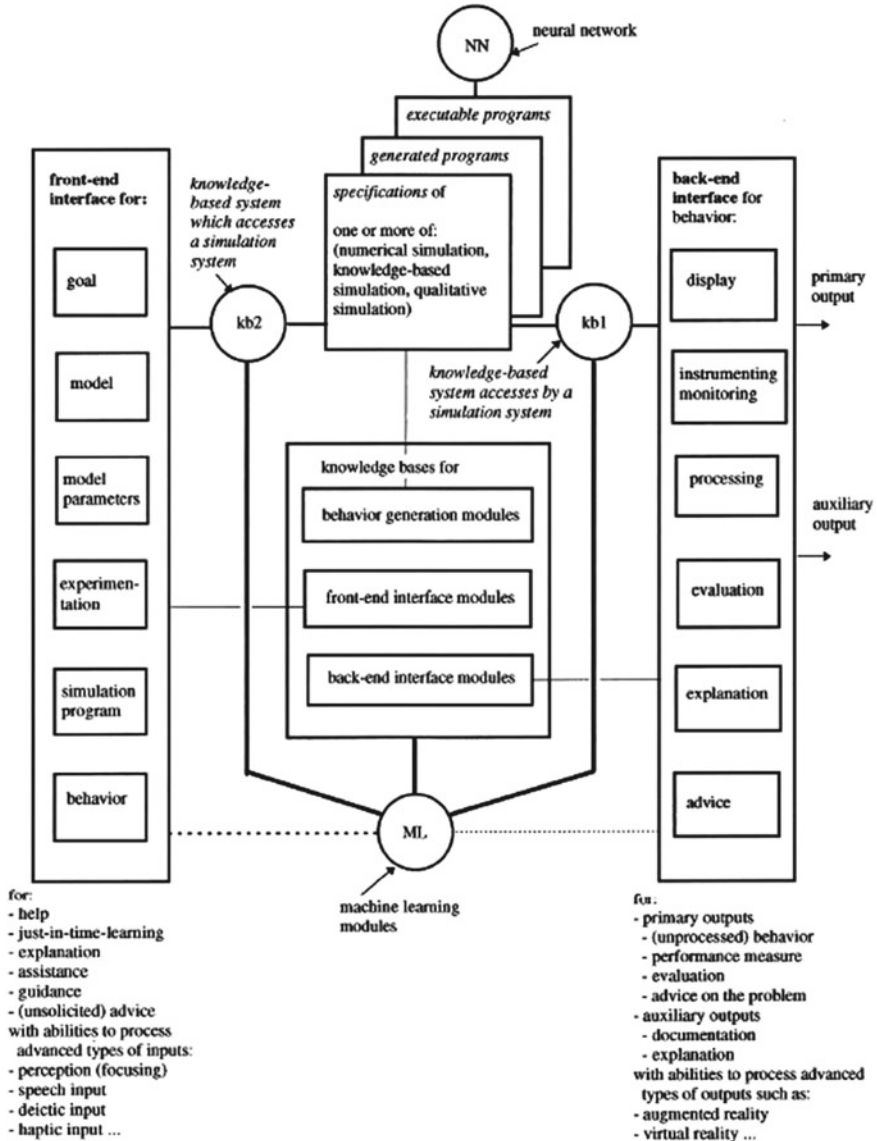


Fig. 12.1 Elements of cognitive simulation environments

12.4 Example: Ant Colony Simulation

Claudia Szabo

The ant foraging model or the ant colony [38] is a widely used model to showcase self-organization and emergent behavior in complex systems research [39]. It is also used extensively in optimization algorithms [40]. In particular, ant colony optimization is used extensively in a variety of domains such as manufacturing, transportation [41], Internet of Things [42] and networking [43] among others.

In the model inspired from real-life ants, agents or ants from an anthill will randomly search for food sources. When a food source is found, ants will bring a unit of food back to the central nest. While returning, the ants will drop pheromones. If an ant is carrying food, it will follow the path of highest pheromone concentration back to the nest.

Fundamental to the ant colony simulation model is pheromone communication, which is widely found in nature and forms the basis of a large number of swarm-based approaches. Pheromones are used by social insects such as bees, ants, and termites both for communication between agents and for communication between the agents and the swarm. Artificial pheromones have been adopted in the multi-robot and swarm robotic domains [44], where it can be implemented in various ways, such as chemical and physical among others. Pheromone-based communication and control are also widely used in swarming UAVs [45, 46].

12.5 Synergies of Agents and M&S

Tuncer Ören

The term “agent” has been used in English since 1610 to mean “one who acts” (OED-agent [32]). In software engineering and artificial intelligence, most often, the term “agent” is used as a short form of “software agent.” “Agents are software modules with cognitive abilities that can work as assistants to the users. They can observe and sense their environments as well as affect it. Their cognitive abilities include (quasi-)autonomy, perception, reasoning, assessing, understanding, learning, goal processing, and goal-directed knowledge processing [47, 48] (Finin et al. 1997).” [49].

The following quotation provides additional clarification about agents:

Agents are autonomous software modules with perception and social ability to perform goal-directed knowledge processing over time, on behalf of humans or other agents in software and physical environments. When agents operate in physical environments, they can be used in the implementation of intelligent machines and intelligent systems and can interact with their environment by sensors and effectors. The core knowledge processing abilities of agents include: reasoning, motivation, planning, and decision making. The factors that may affect decision making of agents, such as personality, emotions, and

cultural backgrounds can also be embedded in agents. Additional abilities of agents are needed to increase their intelligence and trustworthiness. Abilities to make agents intelligent include anticipation (pro-activeness), understanding, learning, and communication in natural and body language. Abilities to make agents trustworthy as well as assuring the sustainability of agent societies include being rational, responsible, and accountable. These lead to rationality, skillfulness and morality (e.g., ethical agent, moral agent). [50]

Agent-based models (ABM) are a powerful paradigm, and use of agent-based models is widespread and important. Lists of ABM researchers and ABM resources (including organizations, centers, and institutes; tools; and tutorials) can be found at (ABM researchers [51] and ABM resources [52], respectively).

Holons are special types of agents with ability to cooperate. “Holon systems are excellent candidates to conceive, model, control, and manage dynamically organizing cooperative systems. A *holonic* system is composed of autonomous entities (called *holons*) that can deliberately reduce their autonomy, when need arise, to collectively achieve a goal. A *holonic* agent is a multi-agent system where each agent (called a holon) acts with deliberately reduced autonomy to assure harmony in its cooperation in order to collectively achieve a common goal. ... *Holon agent simulation or holon simulation*, in short, is an important type of agent simulation where agents represent holons. Some military application include use of simulation for preparedness for conflict management including conflict avoidance, conflict resolution, and conflict deterrence. Civilian applications include modelling and simulation of cooperation of different business entities.” [53]. In recent publications, ethical considerations in cooperation were elaborated on and conditions when cooperation is not desirable from an ethical point of view are clarified [54]. Ethical holon cooperation opens new vistas in artificial intelligence for cooperation of autonomous and quasi-autonomous systems.

Agents provide a powerful computational paradigm. Appendices 1 and 2, adapted from (Ören [53] and TBD-dic [55]), provide lists of types of agents and agent-related concepts, respectively.

A meaning of the term “synergy” is “a mutually advantageous conjunction or compatibility of distinct business participants or elements.” (M-W-synergy). Both experiment and experience aspects of simulation and agents do contribute to each other. As seen in Fig. 12.2 (adapted from [50], synergies of agents and modeling and simulation, also named agent-directed simulation (ADS), consist of: (1) contributions of simulation to agents and (2) contributions of agents to simulation that can be shortened as simulation for agents and agents for simulation.

Contributions of simulation to agents, as outlined in Sect. 12.4.1, are agent simulation which is widely used for the simulation of systems modeled as agents.

Contributions of agents to simulation involve two classes of possibilities: agent-monitored simulation and agent-supported simulation. They are clarified in the following sections of this chapter in more detail.

Historically, agent-directed simulation is started with the collaboration of Ören and Yilmaz. A list (covering four decades) of over 80 publications and over 60 meeting activities of Ören and his colleagues on software agents and agent-directed

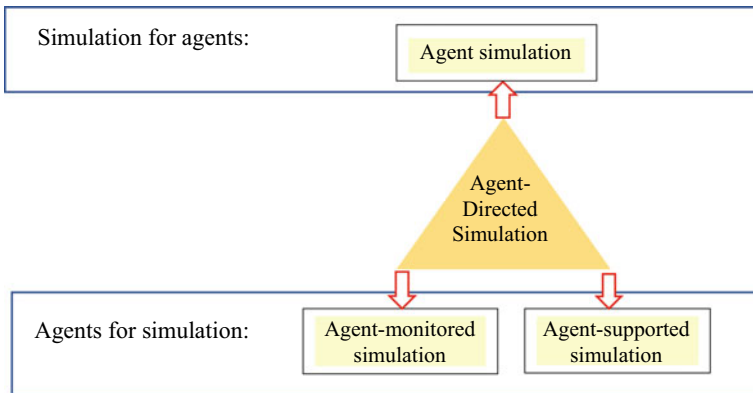


Fig. 12.2 Synergies of agents and modeling and simulation, also named agent-directed simulation (ADS) (adapted from [50])

simulation can be found at Ören-agents [32]. Yilmaz’s contributions to ADS can be seen at (Yilmaz-CV).

As previously mentioned, when cooperation is concerned, holonic agents may become important in all aspects of agent-directed simulation and may open the door for Holon-directed Simulation (HDS).

A Historic Note

Ándras Jávör did pioneering work by using “demons” in modeling and simulation. In those early days, even the term “software agent” was not yet introduced in the scientific literature [56–58]. Other early contributors to the field are Hesper and Hogeweg [59]. For an additional overview, the interested reader is referred to [60].

12.6 Facets of Agent-Directed Simulation

Valdemar Vicente Graciano Neto, Tuncer Ören

12.6.1 Agent Simulation or Agent-Based Simulation

In artificial intelligence (AI), an agent refers to an autonomous entity that acts toward achieving goals, using observation through sensors and actuating upon an environment [61]. Agents are often engineered to be governed by internal rules that drive how they interact and react to its surrounding environment or its representation. Due to their appealing nature, agents are recurrent technologies used to represent multiple independent entities interacting among themselves and with the environment, such as in avatars in virtual games, social interaction studies, or swarms.

A study that analyzed 32,576 studies on agents enables to affirm that agents and computer-based simulation have almost the same age, both with seminal studies in 1950s and 1960s [62, 63]. Indeed, agent design and simulation are congruent sub-domains of agent-based computing [63].

Several tools have emerged to support an agent-based simulation, i.e., the conception of simulation models based on the same idea of agents as individual autonomous entities that can interact to support a better understanding of global or emergent phenomena associated with complex adaptive systems [63].

The synergy of simulations and agents enabled visual perceptions of agents' interaction and supported visualization of emergent structures, such as in NetLogo, which provides agent-based simulation and mechanisms for visualization.

Moreover, it is not rare to find studies that combine agent-based simulation and DEVS, for instance [64].

12.6.2 Agent-Monitored Simulation

Agent-monitored simulation is a powerful possibility for advanced simulation methodologies and consists of roles agents can play during simulation run-time.

An example on agent-monitored simulation is agent-monitored anticipatory multisimulation: a systems engineering approach for threat-management training [65].

Multisimulation with multimodels, multiaspect models or multistage models needs mechanisms to decide when and under what conditions to replace existing models with a successor or alternative. The control agent, shown in Fig. 12.3, monitors the multisimulation subsystem through the anticipatory learning component. The control agent partly enables branching into contingency plans and behavioral rules in response to scenario or phase change during experimentation. Graphs of model families may facilitate derivation of feasible sequence of models that can be invoked or staged. More specifically, a graph of model families can be used to specify alternative staging decisions. Each node in the graph depicts a model, whereas edges denote transition or switching from one model to another. [65].

Another example for agent-monitored simulation is the role of agents in dynamic model couplings. Model coupling allows a robust way of model composition from component models (some of which can be already coupled models). Clarification about model coupling can be found at [65–68]. In dynamic couplings, there may be two types of time or state-dependent changes: (1) Input/output relationships of models may be updated during run-time, (2) some models (or multi-models) may be replaced by some others, or (3) both I/O relations and models may be updated. Agents are most adequate for monitoring condition to implement dynamic couplings.

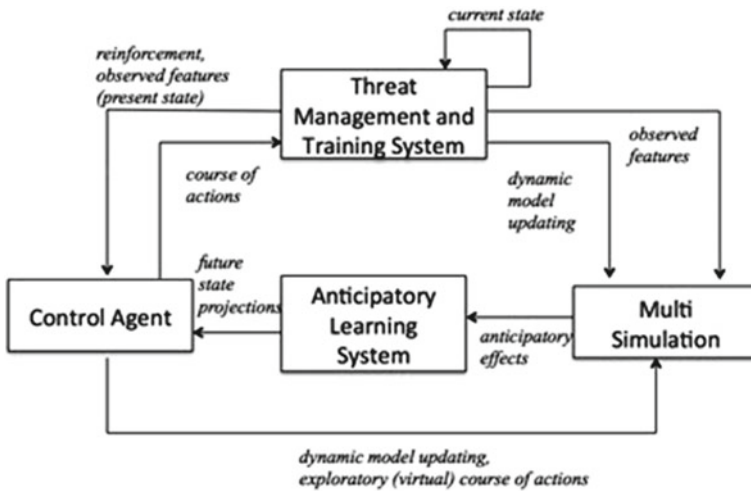


Fig. 12.3 Anticipatory learning in threat management (from Ören and Yilmaz [65])

12.6.3 Agent-Supported Simulation

Agent-supported simulation “is the use of agent technology to support modeling and simulation activities as well as simulation-based problem solving environments (or simulative problem solving environments).” [60]. Possibilities for agent support in modeling and simulation are outlined in Table 12.3.

In this section, agent support in modeling and simulation is covered in the following categories of important possibilities for advanced M&S:

1. Agent support for front-end interfaces
2. Agent support for back-end interfaces
3. Agent support to provide cognitive abilities to the elements of M&S systems
4. Agent support for symbolic processing of elements of M&S studies
 - 4.1 Agent-supported model processing including model transformation
 - 4.2 Agent-supported experimentation
 - 4.3 Agent-supported interoperability

A reference on agent-supported simulation was developed by Ören et al. [60]. R&D, both at academic and industrial levels, are warranted to explore the benefits of agent-supported simulation.

12.6.3.1 Agent Support for Front-End Interfaces

Table 12.4 “outlines the front-end functionalities for the elements of a modeling and simulation environment.”

Table 12.3 Possibilities for agent support in modeling and simulation (adapted from Ören et al. [60], Ören [69])

Applicable for	For specification/generation/editing of	For processing
Goal	<ul style="list-style-type: none"> • Goal specification/editing • Goal generation • Hypotheses formulation 	<ul style="list-style-type: none"> • Goal processing <ul style="list-style-type: none"> – goal seeking – goal modification – goal evaluation – goal selection
Parametric model	<ul style="list-style-type: none"> • Modeling <ul style="list-style-type: none"> – model composition – model coupling – model editing 	<ul style="list-style-type: none"> • Model-base management • Model analysis <ul style="list-style-type: none"> – model characterization – model evaluation • Model transformation
Parameters of <ul style="list-style-type: none"> – models – experimentation/experiences 	<ul style="list-style-type: none"> • Parameter estimation/calibration • Editing <ul style="list-style-type: none"> – parameters – auxiliary parameters 	<ul style="list-style-type: none"> • Symbolic processing <ul style="list-style-type: none"> – parameters – auxiliary parameters
Design of experiments	<ul style="list-style-type: none"> • Design/editing of experiments 	<ul style="list-style-type: none"> • Processing of design of experiments
Every experiment	<ul style="list-style-type: none"> • Specification/editing of experimental conditions (experimental frames) <ul style="list-style-type: none"> – initial conditions of state variables – behavior generator 	<ul style="list-style-type: none"> • Automatic selection of <ul style="list-style-type: none"> – behavior generator – behavior generation

Table 12.4 Some front-end interface functionalities in M&S environments (adapted from Ören et al. [60])

- Anticipation of user’s needs
- Help formulate/specify problems
- Methodology-based model specification
- Awareness, just-in-time-learning, explanation
- Assistance, guidance, (un)solicited advice
- Abilities to process advanced types of inputs:
 - perception (focusing)
 - speech input
 - body language, deictic input, haptic input
 - holographic input
 - thought-control input

12.6.3.2 Agent Support for Back-End Interfaces

“Back-end interfaces are used by systems to communicate to the users the primary and auxiliary outputs of the system. Table 12.5 outlines the back-end functionalities for the elements of modeling and simulation environments.

Back-end interface functionalities provide support for behavior display, instrumenting/monitoring, processing, evaluation, and advice. Advanced types of outputs such as augmented/enhanced reality and virtual reality are part of the

Table 12.5 Some back-end interface functionalities in M&S environments (adapted from Ören et al. [60])

- Primary outputs
- Unprocessed behavior
 - behavior processing
 - statistical
 - analytical
 - visualization
 - performance measure
 - evaluation
 - advice on the problem
- Auxiliary outputs
 - automated documentation
 - explanation
- With advanced abilities to process outputs such as:
 - virtual reality
 - augmented reality
 - holographic visualization

functionalities of back-end interfaces. Back-end interface functionalities are applicable to behavior displays, instrumenting, processing, evaluating, explanation, and warning/advice.” [60].

12.6.3.3 Agent Support to Provide Cognitive Abilities to the Elements of M&S Systems

This is distinct from simulation of systems modeled by agents (or agent simulation).

Software agents can provide cognitive abilities such as perception [70], anticipation [65], and understanding (Ören, Ghasem-Aghaee, and Yilmaz, 2007) to the elements of simulation studies. Table 12.6 outlines some additional possibilities.

12.6.3.4 Agent Support for Symbolic Processing of Elements of M&S Studies

Intelligent agents can provide support in various stages of the M&S process. “For instance, in Model-Driven Engineering (MDE) that involves automated transformation of platform-independent abstract models, agents can serve as transformation engines, by which increasingly concrete and platform-dependent models and simulations can be generated.

Agents can also function as mediators and brokers for distinct simulations by bridging the syntactic and semantic gap between their representations. To support goal-directed experimentation, agents can bring transparency to the overall experiment design [70], execution, analysis, and adaptation process for various types of experiments such as sensitivity analysis, variable screening, understanding, optimization, and decision-support.” [60].

In this section, the following possibilities are outlined:

- Agent-Supported Model Processing including Model Transformation
- Agent-Supported Experimentation.
- Agent-Supported Interoperability.

Table 12.6 Some possibilities to add cognitive abilities for the elements of M&S studies (adapted from Ören et al. [60])

- Cognitive abilities to the elements of M&S systems such as perception, anticipation, understanding, learning, and/or hypothesis formulation
- Program understanding for documentation and/or maintenance purposes
- Agents in simulation-based problem solving environments
- Holons for goal-directed cooperation and collaboration (including “principled holons” who can refuse certain types of cooperation)
- Simulation-based predictive displays for social and financial systems:
 - to train future policy/decision makers
 - to predict abnormal deviations and
 - to test and select possible corrective actions
- Auto-simulation to test and evaluate autonomous decisions by agents
- Agent-based ubiquitous (mobile) simulation (including agent-based mobile cloud simulation)
 - selection of models
 - selection of matching scenarios for experimentation

Agent-Supported Model Processing including Model Transformation

“The common strategy in MDE is based on the application of a sequence of transformations starting with platform-independent models down to the concrete realization of the simulation system. Besides the reuse of models and deployment of designs in alternative platforms, MDE improves the reliability of simulation systems through correctness preserving transformations that allow substantiating the accuracy of realizations with respect to explicit constraints and assumptions defined in the abstract models. ... An agent with understanding capabilities as presented in Ören et al. (2007) can be used to map constructs of a source meta-model to equivalent features of the target meta-model.” [60].

Agent-Supported Experimentation

“An agent-coordinated support system could greatly enhance the experimental design process in several ways, but mainly by providing expert knowledge that the user might lack [70].” [60].

Agent-Supported Interoperability

“In distributed simulation, interoperability refers to the ability of system models or components to exchange and interpret information in a consistent and meaningful manner. This requires both syntactic and semantic congruence between systems either through standardization or mediators that can bridge the syntactic and semantic gap between peer components. Such mediator or bridge agents ensure both data-level interoperability (i.e., metadata/data interchange) and operational-level interoperability (i.e., behavioral specification). Common strategies for

developing adapters involve the provision of standard APIs and connecting components through published interfaces that explicitly specify the required and provided services. These low-level connectors are often limited and do not achieve full data interoperability.” [60].

Appendix 1: Types of agents (adapted from Ören [53] and TBD-dic [55])

Adaptive agent	Competent agent	Dispatched agent
Advertising cookie	Competitive agent	Dispatched mobile agent
Agent	Complete agent	Distant agent
Agent-based holon	Computational agent	Distinguished agent
Animated agent	Computer interface agent	Domain-specific agent
Antagonistic agent	Computer-controlled bot	Emotional agent
Anticipatory agent	Contractee agent	Endomorphic agent
Application agent	Contractor agent	Essential cookie
Artificial moral agent	Conventional agent	Ethical agent
Authorized agent	Conventional software agent	Fixed agent
Autistic agent	Conversational agent	Functional cookie
Autodidactic agent	Cookie	Global agent
Autonomous agent	Cooperating agent	Goal-directed agent
Autoprogrammable agent	Cooperation agent	Goal-oriented agent
Believable agent	Coordination agent	Holonic agent
Bot	Coordinator agent	Independent agent
Broker	Coupled multiagents	Individual agent
Broker agent	Deceptive agent	Information agent
Client agent	Deleted cookie	Information disseminating agent
Cognitive agent	Deliberative agent	Information filtering agent
Co-located agent	Diagnosis agent	Information gathering agent
Co-located agent	Digital agent	Information spider
Communication agent	Disabled cookie	
Intelligent agent	Persistent cookie	Stationary agent
Inter-agent	Personal agent	Subagent
Interface agent	Personal digital agent	Subordinate agent
Intermediate agent	Personal software agent	System latency agent
Internet agent	Proactive agent	Task-specific agent
Itinerant agent	Purposeful agent	Teachable agent
Knowledge-based agent	Rational agent	Temporary cookie
Learning agent	Reactive agent	Tightly coupled multiagent

(continued)

Local agent	Reliable agent	Tracking cookie
Long-lived agent	Remote agent	Transient agent
Loosely coupled multiagent	Resident agent	Transportable Information agent
Mail agent	Retrieval agent	Trusted agent
Marketing cookie	Root agent	Trustworthy agent
Message transfer agent	Rule-based agent	Unauthorized agent
Messaging agent	Scriptable agent	Understanding agent
Mobile agent	Search agent	Uniform resource agent
Model-based agent	Self-motivated agent	User agent
Multiple mobile agent	Self-replicating agent	User interface agent
Network agent	Semantic agent	User-programmed agent
Neural net agent	Semi-autonomous agent	Virtual agent
Notification agent	Service agent	Vivid agent
Pedagogical agent	Session cookie	Wanderer
Permanent cookie	Sociable agent	Web search agent
Permanent cookie	Software agent	Web site agent
Persistent cookie	Spider	Web-oriented agent

Appendix 2: Agent-related concepts (adapted from Ören [53] and TBD-dic [55])

Agency	Agent development platform	Agent software
Agent architecture	Agent efficiency	Agent system
Agent autonomy	Agent framework	Agent understanding
Agent behavior	Agent implementation	Agent user
Agent class	Agent interactivity	Agent-assisted workflow support
Agent code	Agent language	Agent-based
Agent communication language	Agent model	Agent-based adaptive mechanism
Agent communication protocol	Agent security	Agent-based adaptive system
	Agent service	
Agent-based assistant	Agentive	Multiagent learning system
Agent-based cloud computing	Agent-monitored	Multiagent learning technique
Agent-based cognitive architecture	Agent-oriented	Multiagent software
Agent-based complex system	Agent-oriented methodology	Multiagent system
Agent-based complex system development	Agent-oriented modeling	Multiagent understanding
Agent-based design	Agent-oriented problem solving	Multiagent understanding system

(continued)

Agent-based fault-tolerant system	Agent-oriented programming	Ontology-based agent service
Agent-based interaction protocol	Agent-oriented requirements engineering	Privacy in agent-based systems
Agent-based interface	Agent-oriented tool	Safety in agent-based systems
Agent-based knowledge discovery	Agentry	Security in agent-based systems
Agent-based marketplace	Agent-supported	Self-adaptation in multiagent systems
Agent-based model	Animated agent technology	Self-adaptation via multiagent systems
Agent-based modeling	Autonomous agent-based technique	Semantic agent system
Agent-based modeling-as-a-service	Cookie policy	Service-oriented agent-based architecture
Agent-based social simulation	Cookie preference	Service-oriented agent-based protocol
Agent-based software	Ethics for agents	Subagency
Agent-based software engineering	Holonic agent simulation	Task execution in multiagent systems
Agent-based software provider	Intelligent agent modeling	Task planning in multiagent systems
Agent-based system	Intelligent agent system	Task-oriented agent-based system
Agent-based system application	Intelligent agent technology	
Agent-based technique	Inter-agent communication	
Agent-based trust model for cooperation	Inter-agent communication language	
Agent-based ubiquitous service	Inter-agent knowledge processing	
Agent-based ubiquitous system	Learning via multiagent system	
Agent-based virtual enterprise	Lifetime of cookies	
Agent-directed	Mobile agent paradigm	
Agented	Multiagent architecture	
Agent-enabled	Multiagent design-system	
Agent-enabled feature	Multiagent intelligent system	
Agential		

References

1. Zadeh LA (1965) Fuzzy sets. *Inf Control* 8:338–353
2. Zadeh LA (1983) The role of fuzzy logic in the management of uncertainty in expert systems. *Fuzzy Sets Syst* 199–227
3. Dubois D, Prade H (1993) Fuzzy set in approximate reasoning, part 1. *Fuzzy Set Syst* 40
4. Dubois D, Fargier H, Fortin J (2004) A generalized vertex method for computing with fuzzy intervals. In *Proceedings of the international conference on fuzzy systems, IEEE edn. IEEE, Budapest, Hungary*, pp 541–546
5. Zadeh LA (1988) Fuzzy logic. *Computer* 21(4):83–93

6. Saleem K (2008) Fuzzy time control modeling of discrete event systems. In: Proceedings of the World Congress on Engineering and Computer Science, International Association of Engineers (IAENG), pp 683–688
7. Son M-J, Kim T-W (2012) Torpedo evasion simulation of underwater vehicle using fuzzy-logic-based tactical decision making in script tactics manager. *Expert Syst Appl* 39(9):7995–8012
8. Bisgambiglia PA, Capocchi L, Bisgambiglia P, Garredu S (2010) Fuzzy inference models for discrete event systems. In: 2010 IEEE international conference on Fuzzy Systems (FUZZ), pp 1–8
9. Bisgambiglia P-A, Capocchi L, de Gentili E, Bisgambiglia P (2007) Manipulation of incomplete or fuzzy data for DEVS-based systems. In: International modeling and simulation multiconference (IMSM)—conceptual modeling simulation (CMS), pp 87–92
10. Bisgambiglia P-A, de Gentili E, Santucci J, Bisgambiglia P (2006) DEVS-Flou: a discrete events and fuzzy sets theory-based modeling environment. In: Systems and Control in Aerospace and Astronautics, (ISSCAA), pp 95–100
11. Kwon Y, Park H, Jung S, Kim T (1996) Fuzzy-Devs formalism: concepts, realization and application. In Proceedings of AIS, pp 227–234
12. I. E. C. technical committee, Industrial process measurement and control, IEC 61131—programmable controllers, Tech. Rep., 2000, part 7: Fuzzy control programming. IEC
13. Kelber J, Triebel S, Pahnke K, Scarbata G (1994) Automatic generation of analogous fuzzy controller hardware using a module generator concept. In: Proceedings of 2nd European congress on intelligent techniques and soft computing, 8 pp
14. Nhivekar G, Nirmale S, Mudholkar R (2013) A survey of fuzzy logic tools for fuzzy-based system design, vol ICRTITCS, no 9, February 2013, pp 25–28, published by Foundation of Computer Science, New York, USA
15. Umamo M, Mizumoto M, Tanaka K (1978) FSTDS system: a fuzzy-set manipulation system. *Inf Sci* 14(2):115–159
16. Fellinger WL (1978) Specification for a fuzzy system modelling language. PhD dissertation, Oregon State University, Corvallis
17. Alsmadi MKS, Omar KB, Noah SA (2009) Back propagation algorithm: the best algorithm among the multi-layer perceptron algorithm. *Int J Comput Sci Netw Secur (IJCSNS)* 9(4):378–383
18. Maglogiannis IG (2007) Emerging artificial intelligence applications in computer engineering: real word AI systems with applications in Ehealth, Hci, information retrieval and pervasive technologies. *Frontiers in artificial intelligence and applications*, vol 160. Ios PressInc
19. Wilamowski BM (2011) How to not get frustrated with neural networks. In: Proceedings on IEEE international industrial technology (ICIT), pp 5–11
20. Beigy H, Meybodi MR (2000) Adaptation of parameters of BP algorithm using learning automata. In: Sixth Brazilian Symposium on proceedings on neural networks, pp 24–31
21. Sathya R, Abraham A (2013) Comparison of supervised and unsupervised learning algorithms for pattern classification. *Int J Adv Res Artif Intell (IJARAI)* 2(2)
22. Jayalakshmi T, Santhakumaran A (2011) Statistical normalization and back propagation for classification. *Int J Comput Theor Eng* 3(1):1793–8201
23. Ulrich EG, Agrawal VD, Arabian JH (1994) Concurrent and comparative discrete event simulation. Kluwer
24. Zeigler BP, Muzy A, Kofman E (2019) Theory of modeling and simulation, 3rd edn. Academic Press
25. Popovici K, Mosterman PJ (2013) Real-time simulation technologies: principles, methodologies, and applications. CRC Press
26. Capocchi L, Bernardi F, Federici D, Bisgambiglia P-A (2006) Bfs-devs: a general DEVS-based formalism for behavioral fault simulation. *Simul Model Pract Theory* 14(7):945–970

27. Marenus M (2020) Gardne's theory of multiple intelligences, SimplyPsychology, June 9, 2020. <https://www.simplypsychology.org/multiple-intelligences.html>
28. Ören TI (1995-Invited contribution) Artificial intelligence and simulation: a typology. In: Raczynski S (ed) Proceedings of the 3rd conference on computer simulation. Mexico City, November 15–17, pp 1–5
29. Charniak E, McDermot D (1985) Introduction to artificial intelligence. Addison-Wesley, reading, Massachusetts, p 6
30. Symonds AJ (1986) Introduction to IBM's knowledge-systems products. IBM Syst J 25(2):134–146
31. Ören–AISim. Publications, presentations and other activities of Dr. Tuncer Ören on: synergies of artificial intelligence, cybernetics, and simulation. <https://www.site.uottawa.ca/~oren/pubsList/AISim.pdf>
32. Ören-agents. Publications, Presentations and other activities of Dr. Tuncer Ören on: software agents and agent-directed simulation. <https://www.site.uottawa.ca/~oren/pubsList/agents.pdf>
33. Ören T (1985) Intelligence in simulation—editorial. Simuletter—a quarterly publication of SIGSIM, The Special Interest Group on Simulation of the ACM. vol 16, number 1, January, p 3
34. Newell A, Simon HA (1961) The simulation of human thought. In: Current trends in psychological theory. University of Pittsburgh Press
35. Williams RD (ed) (1992). Two approaches to machine intelligence. IEEE Comp 25:78–81
36. Gonzales AJ, Dankel DD (1993) The engineering of knowledge-based systems: theory and practice. Prentice-Hall, Englewood Cliffs, NJ
37. Ören T (1994) Artificial intelligence in simulation. Ann Oper Res 53:287–319. <https://link.springer.com/article/10.1007/BF02136832>
38. Ochoa A, Hernández A, Cruz L, Ponce J, Montes F, Li L, Janacek L (2010) Artificial societies and social simulation using ant colony, particle swarm optimization and cultural algorithms. In: New achievements in evolutionary computation. IntechOpen
39. Bertelle C, Duchamp GH, Kadri-Dahmani H (eds) (2008) Complex systems and self-organization modelling. Springer Science & Business Media
40. Dorigo M, Birattari M, Stutzle T (2006) Ant colony optimization. IEEE Comput Intell Mag 1 (4):28–39
41. Calabrò G, Inturri G, Le Pira M, Pluchino A, Ignaccolo M (2020) Bridging the gap between weak-demand areas and public transport using an ant-colony simulation-based optimization. Transp Res Procedia 45:234–241
42. Said O (2017) Analysis, design and simulation of Internet of Things routing algorithm based on ant colony optimization. Int J Commun Syst 30(8):e3174
43. Ahmed TH (2005, April) Simulation of mobility and routing in ad hoc networks using ant colony algorithms. In: International conference on information technology: coding and computing (ITCC'05), vol 2. IEEE, pp 698–703
44. Payton D, Daily M, Estowski R, Howard M, Lee C (2001) Pheromone robotics. Auton Robot 11(3):319–324
45. Parunak HV, Purcell M, O'Connell R (2002) Digital pheromones for autonomous coordination of swarming UAV's. In: 1st UAV conference, p 3446
46. Van Dyke Parunak, H., Brueckner, S., & Sauter, J. (2002, July). Digital pheromone mechanisms for coordination of unmanned vehicles. In: Proceedings of the first international joint conference on autonomous agents and multiagent systems: part 1, pp 449–450
47. Bradshaw J (ed) (1997) Software agents. AAAI Press
48. Weiss G (ed) (1999) Multi-agent systems: a modern approach to distributed artificial intelligence. MIT Press, Cambridge, MA
49. Ören T (2000) Agent-directed simulation—challenges to meet defense requirements. In: Ören T, Numrich SK, Uhrmacher AM, Wilson LF, Gelenbe E (2000—Invited Paper). Joines JA et al (eds) Agent-directed simulation: challenges to meet defense and civilian requirements.





- Proceedings of the 2000 winter simulation conference, December 10–13, 2000, Orlando, Florida, pp 1757–1762. <https://informs-sim.org/wsc00papers/241.PDF>
50. Yilmaz L, Ören T (2009) Agent-directed simulation, Chap. 4, pp 111–143 of (2009-All chapters by invited contributors). In: Yilmaz L, Ören TI (eds) Agent-directed simulation and systems engineering. Wiley Series in Systems Engineering and Management, Wiley-Berlin, Germany, 520 p
 51. ABM researchers. <http://www.agent-based-models.com/blog/researchers/>
 52. ABM resources. <http://www.agent-based-models.com/blog/resources/>
 53. Ören TI (2001) Advances in computer and information sciences: from abacus to holonic agents. Special issue on artificial intelligence of Elektrik (Turkish J Electr Eng Comput Sci—Published by TUBITAK—Turkish Science and Technical Council) 9(1):63–70. <https://dergipark.org.tr/en/pub/tbtkelektrik/issue/12103/144616>
 54. Ören T, Yilmaz L (2017) The age of the connected world of intelligent computational entities: reliability issues including ethics, autonomy and cooperation of agents. (Invited ebook chapter). In: Nassiri Mofakham F (ed) Frontiers in artificial intelligence—intelligent computational systems. Bentham Science Publishers, pp 184–213
 55. TBD-dic. Turkish informatics society-English-Turkish dictionary. <http://bilisimde.ozenliturkce.org.tr/onerilen-tum-terimler-ingilizce-turkce/>
 56. Jávor A (1990) Demons in simulation: a novel approach, systems analysis, modeling. SIMULATION 7(1990):331–338
 57. Jávor A (1992) Demon controlled simulation, mathematics and computers in simulation vol 34, pp 283–296
 58. Jávor A, Szűcs G (1998) Intelligent demons with hill climbing strategy for optimizing simulation models. In: Summer computer simulation conference, Reno, Nevada, July 19–22, 1998, pp 99–104
 59. Hogeweg P, Hesper B (1979) Heterarchical selfstructuring simulation systems: concepts and applications in biology. In: Zeigler BP, Elzas MS, Klir GJ, Ören TI (eds) Methodology in systems modelling and simulation. North Holland, pp 221–2312
 60. Ören TI, Yilmaz L, Ghasem-Aghaee N (2014) A systematic view of agent supported simulation: past, present, and promising future. Proceedings of the 4th international conference on simulation and modeling methodologies, technologies and applications (SIMULTECH'14), Vienna, Austria, 28–30 August, pp 497–506. (paper nr: 150). Printed in Portugal. ISBN 978-989-758-038-3
 61. Russell SJ, Norvig P (2016) Artificial intelligence: a modern approach. Pearson Education Limited, Malaysia
 62. Zeigler BP (1972) Toward a formal theory of modeling and simulation: structure preserving morphisms. J ACM 19(4):742–764
 63. Niazi M, Hussain A (2011) Agent-based computing from multi-agent systems to agent-based models: a visual survey. Scientometrics 89(2):479
 64. Camus B, Bourjot C, Chevrier V (2015) Combining DEVS with multi-agent concepts to design and simulate multi-models of complex systems (WIP). In: Proceedings of the symposium on theory of modeling & simulation: DEVS Integrative M&S symposium. Society for computer simulation international, pp 85–90
 65. Ören TI, Yilmaz L (2012) Agent-monitored anticipatory multisimulation: a systems engineering approach for threat-management training. In: Breitenacker F, Bruzzone A, Jimenez E, Longo F, Merkurjev Y, Sokolov B (eds) Proceedings of EMSS'12—24th European modeling and simulation symposium, September 19–21, 2012, Vienna, Austria, pp 277–282. ISBN 978-88-97999-01-0 (Paperback). ISBN 978-88-97999-09-6 (PDF)
 66. Ören TI (2014—Invited review paper) Coupling concepts for simulation: a systematic and comprehensive view and advantages with declarative models. Int J Model Simul Sci Comput (IJMSSC) 5(2):1430001–1430017 (article ID: 1430001). <https://doi.org/10.1142/S17939623143000015>

67. Ören T, Mittal S, Durak U (2018) Induced emergence in social system engineering: multimodels and dynamic couplings as methodological bases, Chap. 9. In: Mittal S, Diallo S, Tolk A (eds) (2018) Emergent Behavior in complex systems engineering: a modeling and simulation approach, Wiley. Hoboken, NJ
68. Ören TI, Yilmaz L (2015, Invited article) Awareness-based couplings of intelligent agents and other advanced coupling concepts for M&S. In: Proceedings of the 5th International Conference on Simulation and Modeling Methodologies, Technologies and Applications (SIMULTECH'15), Colmar, France, July 21–23, 2015, pp 3–12
69. Ören TI (1983) Quality assurance of system design and model management for complex problems. In: Wedde H (ed) Adequate modelling of systems. Springer, Heidelberg, pp 205–219. https://doi.org/10.1007/978-3-642-69208-6_31
70. Ören TI (2001) Software agents for experimental design in advanced simulation environment. In: Ermakov SM, Kashtanov YN, Melas V (eds) Proceedings of the 4th St. Petersburg workshop on simulation, June 18–23, 2001, pp 89–95



Supporting Science Areas

13

Bernard P. Zeigler , Paul Wach , Laurent Capocchi ,
Paul Weirich , Mohammad S. Obaidat, Balqies Sadoun,
and Claudia Szabo

Abstract

Science areas supporting modeling and simulation are overviewed in this chapter of the SCS M&S Body of Knowledge. The areas are systems science and engineering, simulation programs for differential equation solution, key features

B. P. Zeigler (✉)
University of Arizona, Tucson, AZ, USA
e-mail: zeigler@rtsync.com

P. Wach
Virginia Tech, Blacksburg, VA, USA
e-mail: paulw86@vt.edu

L. Capocchi
University of Corsica, Haute-Corse, France
e-mail: capocchi@univ-corse.fr

P. Weirich
University of Missouri, Columbia, MO, USA
e-mail: WeirichP@missouri.edu

M. S. Obaidat
University of Texas-Permian Basin, Odessa, TX 79762, USA

King Abdullah II School of Information Technology, University of Jordan, Jordan and
University of Science and Technology, Beijing, China

B. Sadoun
College of Engineering, Al Balqa' Applied University, Al Salt, Jordan

C. Szabo
University of Adelaide, Adelaide, SA, Australia
e-mail: claudia.szabo@adelaide.edu.au

of frequently used distributions in modeling and simulation, queueing theory, characteristics of queuing systems, and statistical tests of hypotheses.

Keywords

Modeling and simulation • Systems science and engineering • Differential equation solution • Distributions • Queueing theory • Statistical tests of hypotheses

13.1 Systems Science

Bernard P. Zeigler, Paul Wach

This chapter briefly reviews systems science and engineering primarily from the perspective of modeling and simulation and its conceptual basis in systems theory as presented in Chap. 1. The Wikipedia website on systems science [1] has an excellent overview of this topic, and we quote the introduction: “Systems science is an interdisciplinary field that studies the nature of systems—from simple to complex—in nature, society, cognition, engineering, technology, and science itself. To systems scientists, the world can be understood as a system of systems [2]. The field aims to develop interdisciplinary foundations that are applicable in a variety of areas, such as psychology, biology, medicine, communication, business management, technology, computer science, engineering, and social sciences.

Systems science covers formal sciences such as complex systems, cybernetics, dynamical systems theory, information theory, linguistics, and systems theory. It has applications in the field of the natural and social sciences and engineering, such as control theory, systems design, operations research, social systems theory, systems biology, system dynamics, human factors, systems ecology, computer science, systems engineering and systems psychology.”

For our purposes, we can divide early systems work into conceptual scheme development and mathematical formalization. Early system theorists naturally focused on conceptual scheme development. Some well-known theorists and their contributions include von Bertalanffy’s General Systems Theory [3], Ashby’s Cybernetics [4, 5] general systems theory. Conceptual scheme development led eventually to attempts to capture the main concepts in mathematical/logical form often with the aim of enabling computerized assistance in dealing with the complexity that systems approaches could engender. Klir’s architecture of systems problem solving [6] and [7] abstract systems are examples.

Perhaps, the first attempt to establish a strong mathematical foundation for systems theory was that of Wymore’s Theory of Systems Engineering [8]. Views of simulation from the perspectives of computational and system-theory-based activities are outlined in Table 2.5 where the influence of Wymore’s system theory on modeling and simulation as a discipline is evident. We include in the category of mathematical systems theory, work in the 1960s that generalized theories from the burgeoning engineering area of mathematical control theory and sought a

“rapprochement” with the logic of computers and automata theory arising from the newly emerging field of computer science. Both control and computer science areas, while hosted by separate research and application communities, have a set of common concepts such as inputs, outputs, states, state transition functions, and input/output time functions, that can be captured by general systems theory. Such rapprochement was the basis for the mathematical systems theory developed by Kalman et al. [9]. As in much of the early development, the latter was developed independently of Wymore’s earlier work and yet related to many of the same concepts.

Wymore [10] Model-Based Systems Engineering can be credited with coining the term MBSE [11, 12], but so far has received minimal attention in the systems engineering community. The related concept of Model-based Simulation is reviewed in Sect. 17.1.5.

Wymore’s *tricotyledon theory of system design* (T3SD) was created “to provide the system theoretic foundations necessary to the study and practice of systems engineering” and “to explicate mathematical system theory as the basis for the development of models and designs of large-scale, complex systems consisting of personnel, machines, and software” [10]. An explicit connection between the concepts of T3SD and the Modeling and Simulation Framework (see Chap. 1) was made by Zeigler [13].

Wach and Salad [14] characterized this connection by representing T3SD as a system coupled to a simulation program as shown in Fig. 13.1. Here, inputs to the system come from the client, the problem, and the design team. Outputs of the T3SD methodology include an input/output (I/O) design specification, performance indexes, test plan, and system design. To provide computational support for determining the performance of the designed system based on the I/O specification, the outputs are coupled with a “simulation program” which generates behavior specified by an I/O relation that is evaluated by the performance indexes under the test plan. In effect, the simulation program transforms these inputs into a model of the proposed technology in virtual (as opposed to actual) form. The modeled technology is then fed back to T3SD for iteration toward improvements needed to achieve the desired performance:

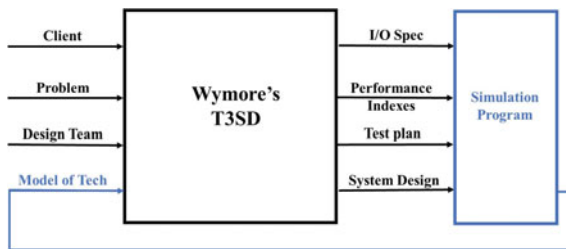


Fig. 13.1 Representation of T3SD operating as a system in conjunction with a simulation program

The workflow of Fig. 13.1 underlies much effort in the systems engineering community in the direction of MBSE and Digital Engineering, its embodiment in computer support [15].

13.2 Mathematics: Differential Equations

Laurent Capocchi

The simulation programs for differential equation resolution have been used since the beginning of the 60's. There are more than hundreds of them, and this is event difficult to have an exhaustive list and/or a classification. However, some of them are well known and have been described in the literature such as EMTF, ECAP, and NETOMAC which have been initiated in the late 60's. Some more recent package such as SPICE (its adaptation to power systems) or MATLAB (its power systems simulator) has been successfully used for many applications. Many of them are adapted to simulation a model defined by sets of ordinary differential equations (ODE) which are related generally to continuous systems under textual or block-diagram representations [16]. During the last twenty years, many efforts have been made in order to give the user modular modeling environments with easy user-based interfaces for transparent and automatic simulation. Nowadays, all the available programs propose graphical interfaces to simplify the modeling process of any physical system. However, these interfaces are more and more complex and can only be fully exploited by experts of the studied domain. Moreover, the simulation algorithms proposed within the programs are mainly based on numerical integration methods such as Runge-Kutta, Euler, Adams, and so on. These methods are specific in the sense that they are all based on a classical discretization resulting in a discrete-time simulation model that could require much more execution time.

An interesting modeling and simulation approach of differential equations resolution using a DEVS-based (Discrete Event system Specification) graphical environment can be proposed. The DEVS formalism [17] has been defined thirty years ago to allow the specification of discrete event systems. It provides a way to define complex models in a hierarchical and modular way. This environment allows the implementation of numerical simulation algorithms based on a discretization which is not linked to time anymore but to space on state variables if the system is continuous. Many years ago, DEVS was upgraded in order to permit the modeling of continuous systems and two main formalisms were developed. In GDEVS (Generalized Discrete Event Specification) [18], the trajectories are organized through piecewise polynomial segments. In the literature, the key contribution of GDEVS is considered as its ability to develop uniform discrete event executable specifications for hybrid dynamic systems. Moreover, discrete event systems including DEVS and GDEVS are simulated at high speed on a host computer because of significant changes in the system. By speaking again of ODE-efficient numerical simulation, another DEVS-based method called quantized systems was

proposed by Zeigler [19]. In this approach, the time discretization is replaced by the state quantization and a DEVS simulation model is obtained instead of a discrete time one. This idea was reformulated and formalized by Kofman [20] as a simulation method for general ODE in where the quantized state system (QSS) method was defined. In [21], the author shows that from the computational cost point of view, the QSS method can reduce the number of iterations. Many examples like block-oriented DEVS simulation of a RLC circuit have been recently presented [22] but the configuration of the QSS integrator model is not explicit. Another example of the circuit simulation based on bond-graph translation was also presented [22]. In [23], Cellier and Kofman give a good reference and strong study of the QSS and its application in the domain of power systems. Recent advances concerning QSS have been published [24–26] and show the power of the QSS method regarding to the classical approach based on time quantization.

13.3 Mathematics: Probability

Paul Weirich

Physical probability and evidential probability play distinct roles in science. Consider physical probability first. An example is the probability that an atom of U_{235} will decay within a year. This probability depends on the physical features of the atom. Another example is the probability of randomly drawing a red ball from an urn with a mixture of red and green balls. This probability depends on the percentage of red balls in the urn.

According to a frequentist interpretation, a physical probability attaches to an event type in a series of trials with outcomes that may belong to the event type, and it equals the limit of the relative frequency of the event type in outcomes of the trials as the number of trials goes to infinity. Hence, the physical probability of getting a red ball from an urn as a result of a random draw with replacement is the limit of the relative frequency of red balls in such random draws as their number goes to infinity. A rival interpretation takes a physical probability to attach to an event and to be the propensity of the event to happen. This physical probability, so interpreted, applies to a single event rather than to a type of event that may recur in a series of trials. It may change as factors influencing the event change. The physical probability of rain on a given day, taken as a propensity, changes as the day's weather conditions change. It may move from 70% in the morning as clouds form to 100% in the afternoon as raindrops begin to fall.

Next, consider evidential probability. The evidential probability of an event is relative to the evidence bearing on the event. At the beginning of a courtroom trial of a defendant accused of committing a crime, the probability of the defendant's guilt may equal 20% given the initial evidence a jury member possesses, but at the end of the trial the probability of the defendant's guilt may equal 80% given the jury member's accumulated evidence. Although the evidential probability of the

defendant's guilt varies with evidence, the physical probability of the defendant's guilt remains constant throughout the trial and is either 0 or 1, depending on whether the defendant committed the crime.

According to the subjective interpretation of evidential probability, the evidential probability of an event is relative to a person's evidence, and two rational people with the same evidence may assign different subjective probabilities to the event. According to the objective interpretation, the evidential probability of an event is relative to a body of evidence and is settled by the body of evidence. If two people with the same evidence attribute different evidential probabilities to the same event, then at least one of them is mistaken about the event's evidential probability.

According to all interpretations of probability, in an algebra of events probabilities comply with three axioms that Kolmogoroff formulated. (1) The probability of every event is non-negative. (2) The probability of a tautology equals 1. (3) The probability that at least one of two exclusive events occurs equals the sum of the probability that the first event occurs and the probability that the second event occurs. Many theorists define the conditional probability of one event given another as the probability of their combination divided by the probability of the second event, the condition. However, some theorists take a conditional probability to be meaningful independently of the ratio of probabilities that this definition uses and hold that a conditional probability equals the ratio because of constraints on conditional probabilities.

For additional information about these topics, the interested reader is referred to the articles [27–31].

13.4 Mathematics: Frequently Used Distributions in Simulation and Their Characteristics and Applications

Mohammad S. Obaidat, Balqies Sadoun

Abstract

Probability distributions are usually proper way to describe real quantities as there is variability in almost any value that can be measured in a system. Furthermore, almost all quantities are made with some inherent error.

In general, a probability distribution is termed discrete if its cumulative distribution function only raises in leaps. It is termed continuous if its cumulative distribution function is continuous. Many probability distributions have been exploited in various applications. Among the popular and most often used distributions/variates are the exponential distribution, normal/Gaussian distribution, Erlang distribution, Weibull distribution, Poisson distribution, geometric distribution, and binomial distribution. In this section, we present main characteristics and applications, the description, and the properties of the probability distributions that are

widely used in modeling and simulation, especially in modeling and simulation computer and network systems.

Probability distributions/variates allow the simulationist to use the right distribution to represent an event or a process based on historical valid experience as this will save a lot of effort and time. The key practical utilizations of the probability distributions/variates include:

- 1 To allow simulation analysis using pseudo-random numbers produced from a particular variate.
- 2 To establish a rational distributional model for the process or phenomenon under study.
- 3 To calculate confidence intervals for parameters and outline crucial areas for hypothesis analyses.

13.4.1 Exponential Distribution

This distribution is very popular in many applications. It establishes a specific essential class of continuous probability distribution [32–37] as it can be employed to model countless systems. The general form of the probability density function (pdf) of an exponential distribution has the form given below:

$$f(t, \lambda) = \begin{cases} \lambda e^{-\lambda t}, & t \geq 0 \\ 0, & t \leq 0 \end{cases}$$

where $\lambda > 0$ is a parameter of the distribution, usually called the rate parameter. This distribution is defined in the interval 0 to ∞ . If X is a random variable that is exponentially distributed, then the Cumulative Distribution Function (CDF) is the integration of pdf and is given by:

$$F(x, \lambda) = \begin{cases} 1 - \lambda e^{-\lambda t}, & x \geq 0 \\ 0, & x \leq 0 \end{cases}$$

Fig. 13.2 depicts the pdf ($f(x)$) and the CDF($F(x)$) for the exponential distribution for various values of the parameter λ . This distribution is regularly used to model the time between independent events that occur at a constant mean rate. It can be used to model systems, processes, and phenomena where an element, initially in state s_0 , then moves to state s , at time t , with an unceasing probability per unit time λ . Thus, the integral from 0 to t of the exponential distribution function is the probability that the system is in state s at time t . In practical scenarios, the hypothesis that λ is constant is hardly met, but in different condition λ can be supposed constant in an interval of time. For instance, the frequency of incoming telephone calls varies based on the time of day, however, if we concentrate on a time interval in which the rate is nearly constant, like around a peak hour, then exponential distribution may be considered as a reasonable estimated model for the time until the subsequent call comes.

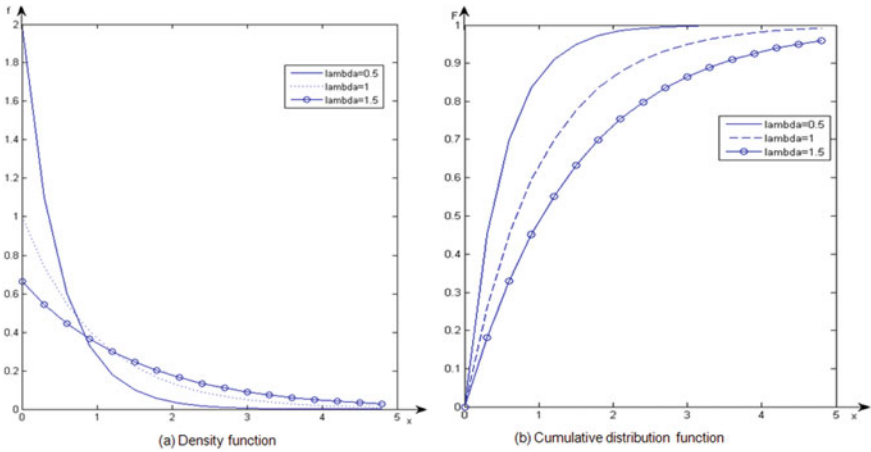


Fig. 13.2 Probability density function and cumulative distribution function of the exponential distribution

In general, exponential variates may be considered to model the times between customers entering the system, nevertheless, the interval of a process that can be considered as a series of several autonomous tasks is better modeled by a variable obeying the sum of a quite a few independent exponentially distributed variables. Furthermore, exponential distribution is suitable for reliability studies as it is very easy to augment failure rates in a reliability paradigm. However, the exponential distribution is not appropriate to model the total lifetime of systems as “failure rates” are not constant. Besides, more failures are expected to occur at the start and end of life phase of a system.

Key Properties of Exponential Distributed Variables

The expected value and variance of an exponentially distributed random variable X with rate parameter λ are given as follows:

$$E(X) = \int_0^{\infty} e^{-\lambda t} dt = \frac{1}{\lambda}, \text{ and}$$

$$V(X) = \sigma^2(X) = \int_0^{\infty} \left(e^{-\lambda t} - \frac{1}{\lambda} \right)^2 dt = \frac{1}{\lambda^2}$$

One key feature of the exponential distribution is that its memoryless, which fully characterizes the exponential distribution. This means that if a random variable X is exponentially distributed, then its conditional probability obeys the following relation:

$$P(X > s + t \mid X > s) = P(X > t), \text{ for all } s, t \geq 0$$

where X measures the time to wait until the first arrival of a customer such as a packet in a computer system or telecommunication network and s and t represent real numbers.

The reverse cumulative distribution function (or quantile function) for the exponential variate with parameter p, λ is given by:

$$F^{-1}(p, \lambda) = \frac{-\ln(1-p)}{\lambda}, \text{ for } p \in [0, 1[$$

Suppose that we know that a particular variable X is exponentially distributed, then the likelihood function for λ , given an independent and identically distributed sample $x = (x_1, \dots, x_n)$ obtained from variable X , is given by:

$$\begin{aligned} L(\lambda) &= \prod_{j=1}^n \lambda e^{-\lambda x_j} = \lambda^n e^{-\lambda(x_1 + \dots + x_n)} \\ &= \lambda^n e^{-\lambda n \bar{x}} \end{aligned}$$

Here, \bar{x} is the sample average given by $\bar{x} = \frac{1}{n} \sum_{j=1}^n x_j$.

13.4.2 Normal (Gaussian) Distribution

The normal or Gaussian distribution is a very popular distribution with many applications in various domains including Electrical Engineering, Computer Engineering, and Science as well as Business Administration and Operation Research.

The normal distribution is a two-parameter family of curves. The first parameter, denoted by μ , is the average/mean and the second parameter, designated by σ , is the standard deviation.

The probability density function (pdf) of the normal distribution is given below [32]:

$$f(x, \mu, \lambda) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(x-\mu)^2/2\sigma^2}$$

The Gaussian function $\varphi(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$ is the density function of the “standard” normal distribution; the normal distribution with parameters $\mu = 0$ and $\sigma = 1$.

One of the major applications of the normal distribution is to use it as a continuous estimate to the binomial distribution via the central limit theorem, which asserts that the sum of independent samples from any distribution with finite mean and variance congregates to the normal distribution as the sample size grows to infinity.

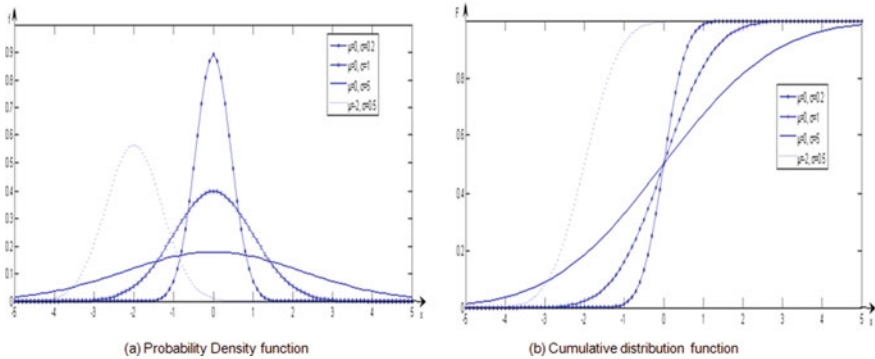


Fig. 13.3 Probability density function and cumulative distribution function of the normal distribution

The Cumulative Distributive Function (CDF) is given as shown below:

$$\begin{aligned}
 F(x, \mu, \sigma) &= \frac{1}{\sqrt{2\pi}\sigma} \int_{-\infty}^x e^{-(t-\mu)^2/2\sigma^2} dt \\
 &= \Phi\left(\frac{x-\mu}{2\sigma}\right)
 \end{aligned}$$

where $\Phi(x)$ is the cumulative distribution function of the standard normal distribution (or $F(x, 0, 1)$) (Fig. 13.3)

$$F(x, 0, 1) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^x e^{-t^2/2} dt$$

The standard normal cumulative distribution function can be conveyed in terms of the Gaussian error function that we signify by $\text{Ger}(-)$:

$$\Phi(x) = \frac{1}{2} \left(1 + \text{Ger}\left(\frac{x}{\sqrt{2}}\right) \right).$$

Function $\text{Ger}(-)$ is defined by:

$$\text{Ger}(x) = \frac{2}{\sqrt{\pi}} \int_0^{\infty} e^{-t^2} dt$$

Here, $\text{Ger}(x)$ cannot be evaluated in a closed form in terms of elementary functions [32–36]; however, it can be expressed in a Taylor series as given below:

$$\text{Ger}(x) = \frac{2}{\sqrt{\pi}} \sum_{n=0}^{\infty} \frac{(-1)^n}{2n+1} \cdot \frac{x^{2n+1}}{n!}$$

Here, the Taylor series converges for every real number x .

Normal distribution has many applications especially modeling errors of any type.

The inverse standard normal cumulative distribution function, or quantile function, can be conveyed in terms of the inverse Gaussian error function $\Phi^{-1}(y) = \sqrt{2}\text{Ger}^{-1}(2y - 1)$. In other words, the inverse cumulative distribution function can be expressed as:

$$F^{-1}(y, \mu, \sigma) = \mu + \sigma\sqrt{2}\text{Ger}^{-1}(2y - 1).$$

The moment generating function is delineated as the expected value of e^{tx} , anywhere this expectation exists. The moment generating function of the normal distribution can be given by:

$$E(e^{tx}) = \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}\sigma} e^{-(x-\mu)^2/2\sigma^2} e^{tx} dx = e^{\mu t + (\sigma^2)t^2/2}.$$

As the moment generating function occurs in an interval around $t = 0$, and $E(e^{tx}) = 1 + \sum_{j=1}^{\infty} t^j m_j$ then the n th moment is given by:

$$\left. \frac{d^{(n)}E(e^{tx})}{dt^n} \right|_{t=0}.$$

In order to use statistical parameters such as average/mean and standard deviation reliably, it is essential to have good estimators of them. Here, the maximum-likelihood estimates (MLEs) offer one such estimator, nevertheless, an MLE may be biased, which means that the expected value of the parameter might not be equal to this parameter. Thus, an impartial estimator that is commonly used to assess the parameters of normal distribution is the minimum variance unbiased estimator (MVUE). The MVUEs of parameters μ and σ^2 for the normal distribution are the sample average and variance.

Assume that X_1, \dots, X_n are independent and normally distributed random variables with mean μ and variance σ^2 . The observed values of these random variables form a sample from a normally distributed population. This is used to estimate the population mean μ and the population standard deviation σ . The joint probability density function of X_1, \dots, X_n is given by:

$$f(X_1, \dots, X_n; \mu, \sigma) = \frac{a}{\sigma^n} \prod_{i=1}^n e^{-(x_i - \mu)^2/2\sigma^2}$$

The prospect function $L(X_1, \dots, X_n)$ is proportional to:

$$\sigma^{-n} e^{-\sum_{i=1}^n \frac{(x_i - \mu)^2}{2\sigma^2}}.$$

The likelihood function is a growing function when the sum $\sum_{i=1}^n (x_i - \bar{x})^2$ diminishes. Then, the maximum likelihood is minimal when this sum is minimal. Let \bar{x} be defined by:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$$

which is the sample mean.

Therefore, the sum $\sum_{i=1}^n (x_i - \bar{x})^2$ is minimized by $\mu = \bar{x}$; that is the maximum-likelihood estimate of μ . Next, we substitute \bar{x} in the likelihood function. The value of σ that maximizes the resulting expression is obtained using the logarithm of the likelihood function, and we have:

$$l(\sigma) = \log \left(L(\bar{x}, \sigma) = cte + -n \log \sigma - \left(\sum_{j=1}^n (x_j - \bar{x})^2 \right) / 2\sigma^2 \right).$$

By applying the derivative to l we obtain:

$$\frac{dl(\sigma)}{d\sigma} = \frac{-n}{\sigma} + \frac{1}{\sigma^3} \sum_{j=1}^n (x_j - \bar{x})^2 = \frac{-n}{\sigma^3} \left(\sigma^2 - \frac{1}{n} \sum_{j=1}^n (x_j - \bar{x})^2 \right).$$

Obviously, it is maximized when $\sigma^2 = \frac{1}{n} \sum_{j=1}^n (x_j - \bar{x})^2$. Consequently, the computed value is maximum-likelihood estimate of σ^2 , and its square root is the maximum-likelihood estimate of σ . This estimator is biased because, if:

$$S^2 = \frac{1}{n} \sum_{i=1}^n (X_i - \bar{X})^2, \text{ where } \bar{X} = \frac{1}{n} \sum_{i=1}^n X_i$$

Then

$$E(S^2) = \frac{n-1}{n} \sigma^2$$

(a) Impartial estimation of parameters: Because the maximum-likelihood estimator of the population mean μ from a sample is an unbiased estimator of the mean, the estimator given below is employed.

$$S^2 = \frac{1}{n-1} \sum_{i=1}^n (X_i - \bar{X})^2$$

So, this is an unbiased estimator of variance σ^2 .

13.4.3 Poisson Distribution

Poisson distribution is a popular discrete probability distribution. One known characteristic of Poisson distribution is that its mean is equal to its variance. It is called after the famous French mathematician Siméon Poisson. It expresses the probability of a number X of events occurring in a fixed period of time if these events occur with a known average rate and are independent of the time since the last event. The Poisson distribution was discovered by Poisson in 1837 when estimating formulas for the binomial distribution considering that the number of trials is high, and the probability of success is little.

It is a one parameter discrete distribution that selects non-negative integer quantities [32–35]. The probability that there are correctly k events is specified by Poisson probability distribution function:

$$f(k, \lambda) = \frac{\lambda^k e^{-\lambda}}{k!},$$

Here, λ is a positive real number that is equal to expected value (mean) and its variate.

Fig. 13.4 shows the Poisson distribution function for different values of Lambda, λ . The expected value of X is given by the expression shown below:

$$E(X) = \sum_{k=0}^{\infty} k \frac{\lambda^k e^{-\lambda}}{k!} = \lambda e^{-\lambda} \sum_{k=0}^{\infty} \frac{\lambda^k}{k!} = \lambda e^{-\lambda} e^{\lambda} = \lambda$$

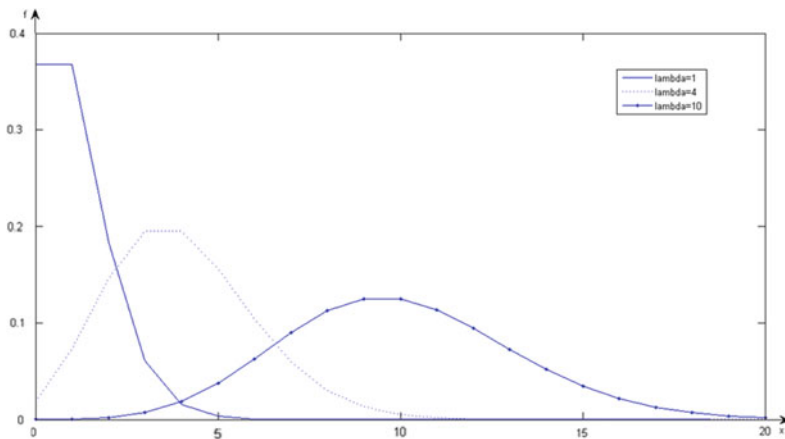


Fig. 13.4 Illustration of Poisson distribution function for different values of Lambda, λ

Furthermore, as indicated earlier, the parameter λ is not only the mean number of occurrences of events, but also its variance, hence, the number of perceived occurrences varies about its mean λ with a standard deviation value of $\sqrt{\lambda}$. Such fluctuations are often called the Poisson noise.

Thus, as the size of the numbers in a specific sample of Poisson random numbers becomes larger, the variability of the numbers will increase as well.

The Poisson distribution has the following main features: (a) Poisson distribution is a discrete distribution, (b) the occurrences in every interval can stretch from zero to infinity, (c) it portrays discrete occurrences over an interval, (d) the average number of occurrences, $E(x)$, must be constant during the experiment, (e) every occurrence is not dependent of the other, (f) it describes the distribution of rare events, and (g) the binomial distribution can be approximated by Poisson distribution as $n \rightarrow \infty$, $p \rightarrow 0$ and np stays constant; hence, a Poisson distribution with $\lambda = np$ strongly estimates the binomial distribution if n is big and p is little. This distribution can be used to represent the distribution of uncommon events in a big population. It is appropriate for purposes that encompass totaling the number of times a random event transpires in a given period of time, distance, and area, among others.

Here are some examples where Poisson distribution can be used: (a) number of viruses that can corrupt a system tied to a network or other systems during a unit of time, (b) number of times a web server is reached per unit time, (c) number of telephone calls at a call center per unit time, (d) number of data packets arriving to a switch per unit time, and (e) number of customers arrive to Hypermart per unit time.

Poisson distribution has special relation with exponential distribution and binomial distribution as explained below:

If the number of counts/arrivals follows the Poisson distribution, then the interval time between individual counts follows the exponential distribution. Thus, the Poisson distribution deals with the number of arrival events in a fixed period of time, and the exponential distribution deals with the time between occurrences of successive events as time flows by continuously.

Poisson distribution is a special case of the binomial distribution as the number of trials, N , is large n and the expected number of successes remains fixed and small. Thus, the Poisson distribution is employed as an estimate of the binomial if n is substantial and p is little.

If λ is the average number of occurrences per unit time and N_t is the number of occurrences before time t , then we can write:

$$P(N_t = k) = f(k, \lambda t) = \frac{(\lambda t)^k e^{-\lambda t}}{k!}.$$

Here, the waiting time X until the first event/occurrence is a continuous random variable with an exponential distribution (with parameter λ); hence, the probability distribution for X can be expressed as follows:

$$P(X > t) = P(N_t = 0) = e^{-\lambda t}.$$

If random variables $X_i, i = 1, \dots, n$ follow a Poisson distribution with parameter λ_i and X_i are independent, then the sum $S = \sum_{i=1}^n X_i$ also follows a Poisson distribution whose parameter is the sum given by: $\lambda = \sum_{i=1}^n \lambda_i$.

The moment generating function of the Poisson distribution with expected value λ is expressed as follows:

$$E(e^{tX}) = \sum_{k=0}^{\infty} e^{tX} f(k, \lambda) = \sum_{k=1}^{\infty} e^{tk} \frac{\lambda^k e^{-\lambda}}{k!} = e^{\lambda(e^t - 1)}.$$

Given a sample of n measured values k_i . To approximate the value of the parameter λ of the Poisson population from which the sample was obtained, we can form the log-likelihood function as follows:

$$\begin{aligned} L(\lambda) &= \log\left(\prod_{i=1}^n f(k_i, \lambda)\right) \\ &= \sum_{i=1}^n \log(f(k_i, \lambda)) = \sum_{i=1}^n \log\left(\frac{e^{-\lambda} \lambda^{k_i}}{k_i!}\right) \\ &= -n\lambda + (\log \lambda) \left(\sum_{i=1}^n k_i\right) - \sum_{i=1}^n \log(k_i!). \end{aligned}$$

In order to compute a maximum, we can find the derivative of function L , with respect to λ , and equate it to zero. This gives us:

$$\frac{dL(\lambda)}{d\lambda} = -n + \frac{1}{\lambda} \sum_{i=1}^n k_i = 0$$

Thus, by solving for λ , we get the maximum-likelihood $\hat{\lambda}$ estimate of λ using:

$$\hat{\lambda} = \frac{1}{n} \sum_{i=1}^n k_i.$$

In general, the Poisson distribution is a means that helps to forecast the probability of particular events from occurring when you know how frequently the event has happened. It provides us the probability of a specified number of events occurring in a fixed interval of time. It is valid only for integers on the horizontal axis.

13.4.4 Uniform Distribution

The uniform distribution is a discrete distribution that has a constant probability distribution function between two parameters, called the minimum and the maximum, a and b , respectively.

It signifies a type of distribution in which all outcomes are similarly probable; every variable has the equal probability that it will be the result.

The standard uniform distribution is a special case of the Beta distribution, obtained by setting both of its parameters to 0 and 1, respectively. The Cumulative Distribution Function (CDF) of the uniform distribution is expressed by:

$$F(x, a, b) = \frac{x - a}{b - a} \chi_{[a,b]}(x)$$

Here, $\chi_{[a,b]}$ is the function defined by: $\chi_{[a,b]}(x) = 1 \iff x \in [a, b]$.

Among the key characteristics of the discrete uniform distribution is that if a random variable has n possible values k_1, \dots, k_n that are equally possible, then it has a discrete uniform distribution. Thus, the probability of any outcome k_i is $1/n$. CDF of a discrete uniform distribution is expressed by:

$$F(x, n) = \frac{1}{n} \chi_{\{k_1, \dots, k_n\}}(x)$$

Here, $\chi_{\{k_1, \dots, k_n\}}$ is the function defined by: $\chi_{\{k_1, \dots, k_n\}}(x) = 1 \iff X \in \{k_1, k_2, \dots, k_{n-1}, k_n\}$. Figure 13.5 shows an example of cumulative distribution function (CDF), F , with 10 values given by $\{k_1 = 0, k_2 = 1, \dots, k_9 = 8, k_{10} = 9\}$. Here, CDF or F , is given by:

$$F(x, 10) = \frac{i}{10}, x \in [i - 1, i[$$

One common example of the discrete uniform distribution is tossing a fair die. The probable values of outcome k are 1, 2, 3, 4, 5, and 6. Every time the die is thrown, the probability of a given score is $1/6$.

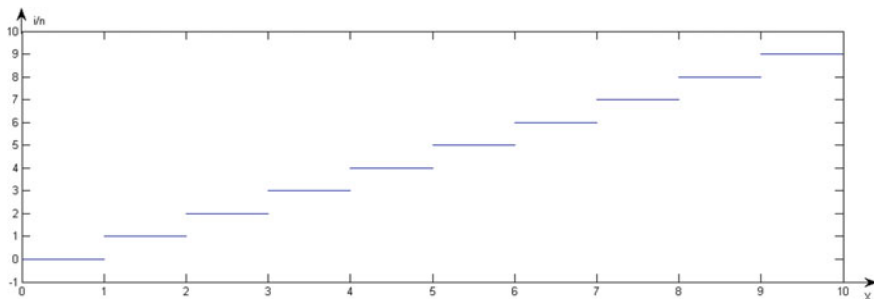


Fig. 13.5 Cumulative Distribution Function (CFDF) of a discrete uniform distribution

This distribution (discrete uniform distribution) can be used to represent random occurrence with several possible outcomes.

As for the continuous uniform distribution, it is one of the easiest distributions to use. It is often used if a random variable is confined, and no additional information is available. Examples include (a) distance between source and destination of message on a computer communication network, (b) seek time on a disk of a computer system, (c) time taken for process to be performed as there are an infinite number of possible times that can be taken, and (d) an individual has an equal probability of drawing a spade, a heart, a club, or a diamond in card dock.

Note that an idealized random number generator is represented by continuous uniform distribution

In order to generate $U(a, b)$, generate $u \sim U(0, 1)$ and return $a = (b - a)u$.

The key parameters for this distribution are:

a = lower limit, b = upper limit, where $b > a$.

13.4.5 Multinomial Distribution

The binomial distribution that has parameters n, p , and k is the distribution of the random variable X , which counts the number of occurrences that happen when n successive data packets are received in a computer network (or a coin is thrown n times), supposing that for any packet, the probability that the packet includes an error (or a head occurs in the case of coin tossing) is p . Here, the distribution function is given by the following formula:

$$P(X = k) = b(n, p, k) = \binom{n}{k} p^k (1 - p)^{n-k}$$

An easy computation reveals that the expected value (mean) and variance of X are equal to np and $np(1 - p)$, respectively.

Binomial distribution is basically a special case of multinomial distribution. The multinomial distribution is the probability distribution of the number of “successes” in n independent Bernoulli trials, where every trial resulting in one of specific fixed finite number k of possible outcomes happening with probabilities p_1, \dots, p_k , and there are n independent tests. If X_i represents the number of times an outcome number i was observed over the n trials, then the multinomial distribution X can be stated as the distribution of the vector (X_1, \dots, X_n) [1,5]. These probabilities can be expressed as follow:

$$P(X = (k_1, \dots, k_n)) = P(X_1 = k_1, \dots, X_n = k_n) \\ = \begin{cases} \frac{n!}{k_1! \dots k_n!} \prod_{j=1}^n p_j^{k_j}, & \text{if } \sum_{j=1}^n k_j = n \\ 0, & \text{otherwise} \end{cases} .$$

Now, every component $X_j, j \in \{1, \dots, n\}$ of random variable X separately has a binomial distribution with parameters n and p_j , and has a mean (expected value) equals to np_j and a variance equals to $np_j(1 - p_j)$. Thus, due to the constraint that the sum of the components is n , then variables are correlated. Here, the covariance matrix $\{\text{Cov}_{i,j}\}_{i,j \leq n}$ is described by: (a) the off diagonal quantities are expressed by $\text{Cov}_{i,j} = \text{cov}(X_i, X_j) = -np_i p_j, i \neq j$, and (b) the elements of the diagonal are expressed by $\text{Cov}_{i,i} = \text{var}(X_i) = np_i(1 - p_j)$.

It is recognized that the Poisson distribution can be employed as an estimate of the binomial distribution when the parameter n is big and p is little. Thus, let us consider a random variable X having a binomial distribution with factors n and p . If we let X count the occurrences of an event in an assumed interval, we can notice λt happenings of an event in a time interval of length t . Thus, if this time interval is allotted into n small intervals, then we must have $\lambda t = np$.

Consequently, we have: $p = \frac{\lambda t}{n}$.

Once computing $P(X = k)$, one can affirm that:

$$P(X = 0) = b(n, p, 0) = (1 - p)^n = \left(1 - \frac{\lambda}{n}\right)^n$$

$$\frac{b(n, p, k)}{b(n, p, k-1)} = \frac{\lambda - (k-1)p}{k(1-p)} \cong \frac{\lambda}{k}, \text{ for large } n \text{ (and, therefore, little values of } p)$$

$$P(X = 1) \approx \lambda e^{-\lambda} \text{ and } P(X = k) \approx \frac{\lambda^k}{k!} e^{-\lambda}, \text{ for large values of } n.$$

As a consequence, one can infer that when n is large, the distribution of X is the Poisson distribution. The multinomial distribution is used to discover probabilities in experiments where there are more than two products.

13.4.6 Log-Normal Distribution

A log-normal or Galton distribution is basically a probability distribution with a normally distributed logarithm. In general, we say that a random variable is log-normally distributed if its logarithm is normally distributed. Thus, if X is a random variable with a normal distribution, then the random variable e^X has a log-normal distribution. This definition is articulate as $\log_a X$ is normally distributed if and only if $\log_b X$ is normally distributed.

The log-normal distribution has the following probability density function, pdf:

$$f(x, \mu, \sigma) = \frac{1}{\sigma \sqrt{2\pi x}} e^{-(\log x - \mu)^2 / 2\sigma^2}, x > 0$$

Here, $\log_e X$ is to base e or $\text{Ln } x$. The expected value and the standard deviation of X are given by:

$$E(X) = e^{\mu + \sigma^2/2}, \text{ and } \sigma^2(X) = (e^{\sigma^2} - 1)e^{2\mu + \sigma^2}.$$

More generally, the k th moment, $k \geq 2$, is given by:

$$m_k(X) = e^{k\mu + k^2\sigma^2/2}.$$

In order to provide the greatest possibility estimators of the log-normal distribution parameters μ and σ , we basically can use the approach applied to the normal distribution. Else, we can observe that the density function f_L of the log-normal distribution and the normal distribution, f_N , are related by the expression below:

$$f_L(x, \mu, \sigma) = \frac{1}{x} f_N(\log x, \mu, \sigma)$$

This, we can write the log-likelihood function $l_L(\mu, \sigma)$ using the log-likelihood function $l_N(\mu, \sigma)$ as shown below:

$$l_L(\mu, \sigma) = \sum_{k \leq n} \left(\log(x_k) + l_N(\mu, \sigma) \right).$$

As the first term is the expression right side is constant with respect to μ and σ , thus, the logarithmic likelihood functions $l_L(\mu, \sigma)$ and $l_N(\mu, \sigma)$ reach their extreme values with the same values of factors μ and σ . Thus, the expressions for the normal distribution maximum-likelihood parameter estimators, that we have previously established, can be employed to infer that:

$$\hat{\mu} = \frac{1}{n} \sum_{k \leq n} \left(\log(x_k) \right) \text{ and } \hat{\sigma}^2 = \frac{1}{n} \sum_{k \leq n} \left(\log(x_k) - \hat{\mu} \right)^2.$$

13.4.7 Weibull Distribution

The Weibull distribution is called after its inventor, Waloddi Weibull. It is being used in many applications especially reliability engineering due to its flexibility and relative cleanness. It is considered as one among the most popular statistical model for life data. Moreover, it is used in many other applications including weather prediction and for fitting data of all types. It can be utilized for traffic engineering analysis with smaller sample sizes.

It is a continuous distribution, and its probability density function, pdf, can be given as below:

$$f(x, k, \lambda) = \begin{cases} \frac{k}{\lambda} \left(\frac{x}{\lambda}\right)^{k-1} e^{-(x/\lambda)^k}, & x \geq 0, k > 0 \\ 0, & x < 0 \end{cases}$$

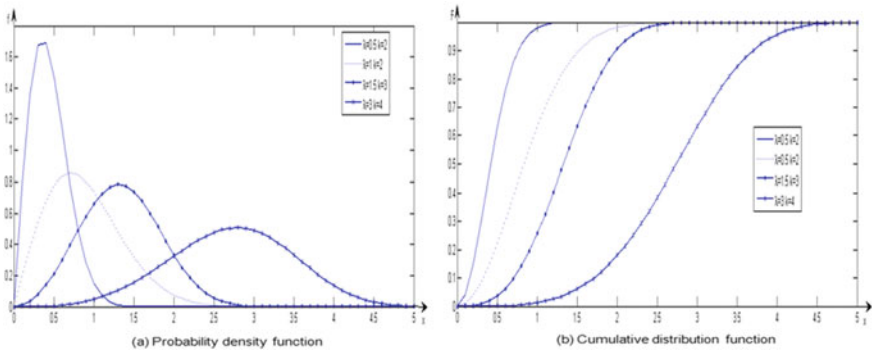


Fig. 13.6 Probability density function and cumulative distribution function of the Weibull distribution

Here, $k > 0$ is called the shape parameter and $\lambda > 0$ is called the scale parameter of the distribution [38].

The Cumulative Distribution Function (CDF) of this distribution is given by:

$$F(x, k, \lambda) = \begin{cases} 1 - e^{-(x/\lambda)^k}, & x \geq 0, k > 0 \\ 0, & x < 0 \end{cases}$$

It is interesting to note that when $k = 1$, then the Weibull distribution becomes Exponential. Obviously, the exponential distribution is a particular case of Weibull distribution. Figure 13.6 shows the pdf and CDF of Weibull distribution for different values of the parameters k and λ .

An important quantity called failure rate in the Weibull distribution is defined by $\frac{kx^{k-1}}{\lambda^k}$. Here, we have three circumstances that can occur: (a) If $k < 1$, then the failure rate decreases over time, (b) if $k = 1$, then the failure rate is constant over time and the distribution turns out to be exponential, and (c) if $k > 1$, then the failure rate increases over time.

In order to show why this definition is made, let us recall that, if $f(t)$ and $F(t)$ are a probability density function (pdf) and its Cumulative Distribution Function (CDF) of t , then the failure rate is given by:

$$h(f)(t) = \frac{f(t)}{1 - F(t)}.$$

By substituting pdf and CDF, the exponential distribution for $f(t)$ and $F(t)$ above produces exactly $\frac{kx^{k-1}}{\lambda^k}$.

The Weibull distribution is frequently used to imitate the behavior of other statistical distributions like the normal and the exponential. Among its main applications are reliability and lifetime modeling. In general, it is suitable to use it to model random failures and multiple source failures, as well as to model the valuable life of products.

Weibull distributions also can be utilized: (a) to model channel fading since the Weibull fading model appears to reveal good fit to experimental fading channel measurements, (b) to model the scattering of the received signals level created in radar systems, (c) to generate statistical model in reliability engineering and breakdown analysis, (d) to describe manufacturing and transfer times in industrial engineering problems, and (e) to represent wind speed distributions and weather predicting models.

This distribution is tightly related the Gamma distribution/function. For instance, we can note that the expected value, n th moment, and standard deviation of random variable X having a Weibull distribution are given by expressions below:

$$\begin{aligned}\sigma^2(X) &= \lambda^2 \cdot \Gamma\left(1 + \frac{2}{k}\right), \\ E(X) &= \lambda \cdot \Gamma\left(1 + \frac{1}{k}\right), \text{ and} \\ m_n &= \lambda^n \cdot \Gamma\left(1 + \frac{n}{k}\right), \text{ respectively.}\end{aligned}$$

13.4.8 Pareto Distribution

This distribution is called after Vilfredo Pareto who is an Italian economist/engineer/sociologist; it is popular in applications in the domain of traditional science, social science, and geography, among others. It can be applied to many situations in communication.

And it is very popular to represent distribution of income in the society.

In general, Pareto distribution can also be applied to many situations in which equilibrium can be found using the distribution. The following examples represent classical examples that sometimes seen as approximately Pareto-distributed: occurrences of words in lengthy texts, file size distribution of communication traffic, and the standardized price returns on individual stocks [32].

If X is a random variable, we say it has a Pareto distribution if there are a positive parameter/factor “ k ” and a positive real value “ a ” such that the probability that X is greater than some number x is expressed by:

$$P(X > x) = \left(\frac{a}{x}\right)^k, x \geq a.$$

The probability density function, pdf, of Pareto distribution is expressed by:

$$f(x, k, a) = k \frac{a^k}{x^{k+1}}, x \geq a, k > 0.$$

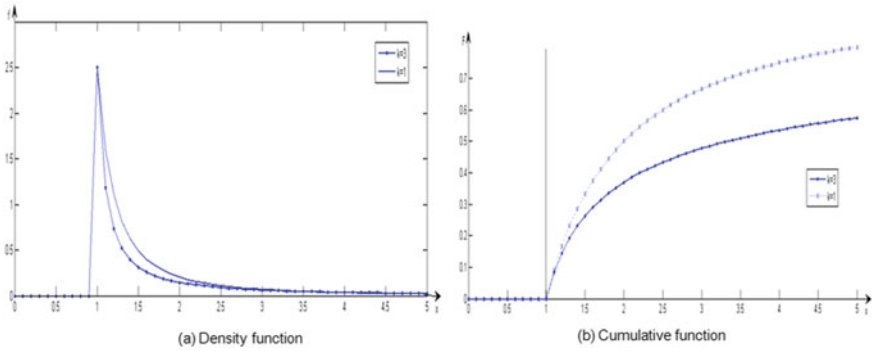


Fig. 13.7 Pareto probability density function (pdf) and its cumulative distribution function (CDF)

Pareto distribution is considered a continuous distribution. The expected value and variance (if $k < 1$) of a random variable X following a Pareto distribution are expressed as follows:

$$E(X) = \frac{ka}{k - 1}, \text{ and } \sigma^2(X) = \frac{k}{k - 2} \left(\frac{a}{k - 1} \right)^2, k < 1.$$

Further, the n th moment of a Pareto random variate, X , is expressed by:

$$m_n(X) = \frac{ka^n}{k - n}.$$

Here, the moments are only described for $k > n$, which means that the moment generating function, which is just a Taylor series, is not well defined. (Fig. 13.7)

Given a sample $x = (x_1, \dots, x_n)$ of a Pareto distribution, the likelihood function $L(k, a)$ for parameters k and a is expressed by:

$$L(k, a) = \prod_{i=1}^n k \frac{a^k}{x_i^k} = k^n a^{kn} \prod_{i=1}^n \frac{1}{x_i^{k+1}}$$

Applying the logarithm function to $L(k, a)$, the logarithmic likelihood function is given by:

$$l(k, a) = \log(L(k, a)) = n \log(k) + nk \log(a) - (k + 1) \sum_{i=1}^n \log(x_i).$$

It can be noticed that the function $\log(L(k, a))$ is monotonically increasing with respect to parameter a . Since $x_i \in [a, \infty[$ for every $1 \leq i \leq n$, we can conclude that the least x_i gives an estimation of parameter a (i.e., $\hat{a} = \min_i x_i$). To find the estimator for k , we calculate the partial derivative with respect to k and equate it to zero as demonstrated below:

$$\frac{\partial l(k, a)}{\partial k} = \frac{1}{k} + n \log(a) - \sum_{i=1}^n \log(x_i) = 0.$$

Consequently, the maximum-likelihood estimator for k is expressed as follows:

$$\hat{k} = 1/\log(a) - \sum_{i=1}^n \log(x_i).$$

The generalized Pareto distribution permits a continuous range of potential shapes that include the Pareto and Exponential distributions. Here, the probability density function, pdf, for the generalized Pareto distribution has three parameters: shape parameters k , location parameters μ , and scale parameter σ . The probability density function, pdf, can be expressed as follows:

$$f(x, k, \mu, \sigma) = \frac{1}{\sigma} \left(1 + k \frac{x - \mu}{\sigma} \right)^{(-1-\frac{1}{k})}.$$

The Cumulative Distribution Function, CDF, can be expressed as follows:

$$F(x, k, \mu, \sigma) = 1 - \left(1 + k \frac{x - \mu}{\sigma} \right)^{(-\frac{1}{k})}, \quad \text{for } x \geq \mu \text{ and } x \leq \mu - \frac{\sigma}{k} \text{ (if } k < 0)$$

Here, when k comes near 0, pdf can be given by:

$$g(x, \mu, \sigma) = \frac{1}{\sigma} e^{-\frac{x-\mu}{\sigma}}$$

Clearly, the generalized Pareto distribution is equivalent to the exponential distribution.

13.4.9 Geometric Distribution

The Geometric distribution is a discrete distribution, described on the non-negative integers. It is considered as an oversimplification of together the exponential and chi-squared distributions. Like the exponential distribution, it is employed for a state-of-the-art model of waiting times. It is useful for modeling the runs of repeated successes (or failures) in recurrent independent trials of a system. The geometric distribution models the number of successes prior to one failure in an independent sequence of tests where each test leads to success or failure. Again, it is considered a special case of the negative binomial distribution and deals with quantity of trials necessary for a single success; hence, the geometric distribution is a negative binomial distribution in which the number of successes (r) is equal to 1.

The geometric distribution probability distribution function, pdf, is usually given by the expression:

$$f(k, p) = p(1 - p)^k$$

The expected value of a geometrically distributed random variable X is $1/p$ and the variance is $\sigma^2(X) = \frac{1-p}{p}$. The Cumulative Distribution Function, CDF, is given by:

$$F(k, p) = P(X > k) = 1 - (1 - p)^k$$

Here, the parameter p can be estimated by equating the expected value with the sample mean. Thus, if we let k_1, \dots, k_n be a sample such that $k_i > 1, i \geq 1$, then p can be estimated by:

$$\hat{p} = \left(\frac{1}{n} \sum_{i=1}^n k_i \right)^{-1}$$

The use of geometric distribution is imperative in the theory of waiting queues. It is frequently assumed that, in each small time slot, either 0 or 1 a new packet arrives to the switch. The probability that a customer/packet arrives is p and that no packet/customer arrives is $q = 1 - p$. Then, the time X till the following arrival has a geometric distribution. The probability that no customer arrives in the next n time slots is given by $P(X > n)$ and it is expressed as follows:

$$P(X > n) = \sum_{j=n+1}^{\infty} pq^{j-1} = q^n p \sum_{j=0}^{\infty} q^j = q^n$$

Similar to the exponential distribution, the geometric distribution is the property of memoryless, which means that if an experiment is duplicated until the first success, then, given that the first success has not yet occurred, the conditional probability distribution of the number of further experiments is not contingent on how many failures have been experienced. For instance, a die that one through does not have a 'memory' of the failures seen. The memoryless property in this context can be stated as shown mathematically below:

$$\begin{aligned} P(X > i+j | X > i) &= \frac{P(\{X > i+j\} \cap \{X > i\})}{P(X > i)} = \frac{P(X > i+j)}{P(X > i)} \\ &= \frac{(1-p)^{i+j}}{(1-p)^i} = (1-p)^j = P(X > j). \end{aligned}$$

The geometric distribution Y is a special case of the negative binomial distribution with $r = 1$. More precisely, if X_1, \dots, X_n are independent geometrically distributed random variables with parameter p , then the random variable $Y = \sum_{j=1}^n X_j$ obeys a negative binomial distribution with parameters r and p .

Conversely, let us consider r and k such that $0 < r < 1$ and $0 < k \leq n$, then the random variable X_k has a Poisson distribution with expected value equals r^k/k . Here, the finite sum: $Y = \sum_{j=1}^n jX_j$ has a geometric distribution choosing values in the set of ordinary integers, N , with mean equals $r/(1-r)$.

13.4.10 Gamma Distribution

The Gamma distribution is a family of continuous probability distributions described by two parameters that represent the sum of k exponentially distributed random variables, each of which has a mean μ .

The gamma distribution is used in reliability work to match failure data, as it is appropriately flexible to manage decreasing, stable, and rising failure rates; however, the Weibull distribution is more commonly used.

The probability density function, pdf, of the gamma distribution can be expressed using the gamma function Γ :

$$f(x, a, b) = \frac{1}{b^a \Gamma(a)} x^{a-1} e^{-\frac{x}{b}}, x > 0, a > 0, b > 0$$

where a is termed the shape parameter and b is termed the scale parameter of the gamma distribution.

The gamma distribution can be employed to model the time till the next n events happen. The Geometric distribution emerges in machine learning as the 'conjugate prior' to a duo distribution.

The gamma function is expressed by:

$$\Gamma(a) = \int_0^{\infty} t^{a-1} e^{-t} dt.$$

Otherwise, another parameterization of the gamma distribution can be used in terms of the shape parameter and a parameter β , called the rate parameter, expressed by $\beta = 1/b$. The two parameterizations are commonly used. Their use is dependent on the nature of the problem to be modeled.

The Cumulative Distribution Function, CDF, of the gamma distribution can be expressed in terms of the gamma function Γ :

$$F(x, a, b) = \int_0^x f(t, a, b) dt = \frac{\gamma(a, \frac{x}{b})}{\Gamma(a)}$$

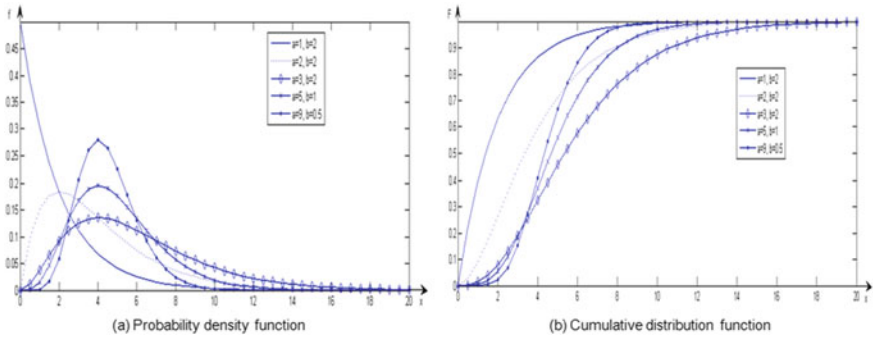


Fig. 13.8 Probability density function and cumulative distribution function of the Gamma distribution

Here, the incomplete gamma function γ [1, 5] is defined by:

$$\gamma(a, x) = \int_0^x t^{a-1} e^{-t} dt.$$

where $\Gamma(a) = \gamma(a, x) + \int_x^\infty t^{a-1} e^{-t} dt, x > 0$. (Fig. 13.8)

If N is an independent and identically distributed random observations (x_1, \dots, x_n) , then the likelihood function related with these observations can be expressed by:

$$L(a, b) = \prod_{i=1}^N f(x_i, a, b).$$

By calculating the logarithm of $L(a, b)$, we can get the log-likelihood function $l(a, b)$ as given below:

$$l(a, b) = \log(L(a, b)) = (a - 1) \sum_{i=1}^N \log(x_i) - \sum_{i=1}^N \frac{x_i}{b} - N(a \log(b) - \log(\Gamma(a))).$$

Thus, by deducting the partial derivative of $l(a, b)$, with respect to b , and equating it to zero, we can obtain the maximum-likelihood estimate \hat{b} of the b parameter. A straightforward computation gives:

$$\hat{b} = \frac{1}{aN} \sum_{i=1}^N x_i.$$

Then by substitution into the log-likelihood function, we can obtain:

$$l(a, \hat{b}) = (a - 1) \sum_{i=1}^N \log(x_i) - \sum_{i=1}^N \frac{x_i}{\hat{b}} - N \left(a - a \log \left(\frac{\sum_{i=1}^N x_i}{an} \right) - \log(\Gamma(a)) \right).$$

This, the maximum of $l(a, \hat{b})$ with respect to a is found by taking the derivative and setting it equal to zero, which gives us:

$$\log(a) - \frac{\Gamma'(a)}{\Gamma(a)} = \log \left(\frac{1}{n} \sum_{i=1}^N x_i \right) - \frac{1}{n} \sum_{i=1}^N \log(x_i).$$

The above expression does not have a closed-form solution as a function of a ; hence, a numerical solution can be found using, for example, the Network's method and beginning with an initial value for a that can be found by using the approximation given below:

$$\log(a) - \frac{\Gamma'(a)}{\Gamma(a)} \approx \frac{1}{2k} + \frac{1}{12k + 2}.$$

Consequently, a can be estimated as follows:

$$a \approx \frac{3 - \bar{s} + \sqrt{(\bar{s} - 3)^2 + 24\bar{s}}}{12\bar{s}}, \text{ Where } \bar{s} = \log \left(\frac{1}{n} \sum_{i=1}^N x_i \right) - \frac{1}{n} \sum_{i=1}^N \log(x_i).$$

13.4.11 Inverse Gamma Distribution

The inverse Gamma distribution is a continuous probability distribution of the reciprocal of a variable distributed according to the gamma distribution. It is used often in Bayesian statistics. It is a two-parameter family of continuous probability distribution that represents the multiplicative inverse of the gamma distribution. The inverse gamma distribution's probability density is expressed over the subset of positive real numbers as follows

$$g(x, a, b) = \frac{b^a}{\Gamma(a)} x^{-a-1} e^{-\frac{b}{x}}, x > 0, a > 0, b > 0$$

Here, a and b are the shape parameter and the scale parameter, respectively. The Cumulative Distribution Function, CDF, is expressed by:

$$G(x, a, b) = \frac{\Gamma(a, \frac{x}{b})}{\Gamma(a)}, \text{ where } \Gamma(a, \frac{x}{b}) = \int_{x/b}^{\infty} t^{a-1} e^{-t} dt$$

where $\Gamma(a, \frac{x}{b})$ is the upper incomplete gamma function.

As an example, consider the issue of examining computer memory ICs and gathering data about their lifespans. We assume that these lifetimes follow a gamma distribution and that we want to find out how long we can anticipate the mean memory chip to live. Here, parameter assessment is the process required to determine the parameters of the Gamma distribution that is appropriate for the setting. We will need to have sample values or observations (x_1, \dots, x_n) , which are the fixed constants. Variables a, b to be found out are the undetermined parameters. Maximum Likelihood Estimation (MLE) deals with finding the values of the parameters that offer the highest likelihood given the specific set of data as described earlier. Typically, the 95% confidence interval can be determined for a and b in order to provide a range of likely values.

13.4.12 Erlang Distribution

The Erlang distribution is a simplification of the exponential distribution. Although the exponential random variable defines the time between contiguous events, the Erlang random variable defines the time interval between any event and the k th next event.

Erlang distribution is a continuous distribution that was devised by Agner Krarup Erlang, a Danish engineer and mathematician who was working in telephone switching and who invented queueing theory and traffic engineering disciplines. He discovered it while studying the number of telephone calls which might be made at the same time to the operators of the switching stations. It has been often used in communication traffic engineering for over 100 years. The Erlang distribution is described by two parameters: an integer k , called the shape; and a real number λ , called the rate [39].

The probability density function, pdf, of the Erlang distribution is expressed by:

$$f(x, k, \lambda) = \frac{\lambda^k x^{k-1} e^{-\lambda}}{(k-1)!}, x > 0.$$

A different parameterization can be performed by replacing in the above expression λ by $1/\theta$, where θ is signified as the scale parameter. The expression shows that the Erlang distribution is only described when the parameter k is a positive integer.

Erlang distribution is considered a special case of the Gamma distribution where the shape parameter k is an integer. As the shape parameter k is 1, the distribution becomes an exponential distribution.

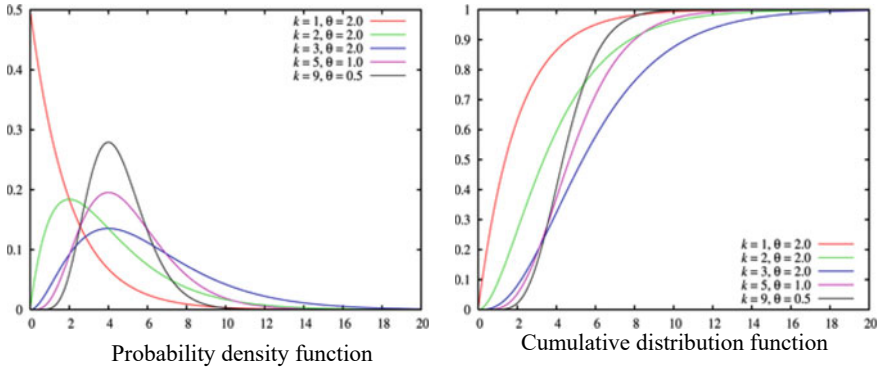


Fig. 13.9 Erlang density and cumulative distribution

The cumulative distribution function, CDF, of the Erlang distribution is expressed by:

$$F(x, k, \lambda) = \frac{\gamma(k, \lambda x)}{(k - 1)!}$$

where $\gamma()$ is the lower incomplete gamma function outlined above. Figure 13.9 depicts the probability density function and cumulative distribution function of the Erlang distribution.

13.4.13 Beta Distribution

Beta distribution represents a group of probabilities that is a flexible way to denote outcomes for proportions. For instance, how likely we will find vaccine for COVID-19 by end of 2020? You might think the probability is 0.7 while your colleague might think it is 0.8. The Beta distribution is a scheme to describe this.

This destruction has two free parameters that are categorized according to one of two notational conventions; the beta probability density function is expressed as follows:

$$f(x, \alpha, \beta) = \frac{1}{B(\alpha, \beta)} x^{\alpha-1} (1-x)^{\beta-1}, B(\alpha, \beta) = \int_0^1 t^{\alpha-1} (1-t)^{\beta-1} dt$$

$$f(x, \alpha, \beta) = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1} (1-x)^{\beta-1}$$

Here, Γ is the gamma function [32, 40]. The expected value and variance of a beta random variate X with parameters α and β are expressed as below:

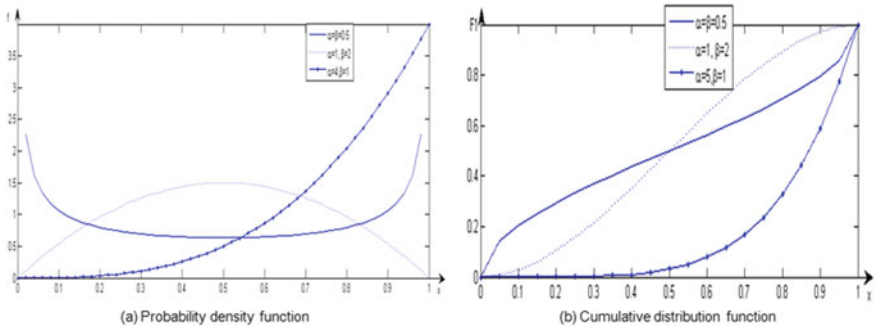


Fig. 13.10 Beta probability density function and cumulative distribution function

$$E(X) = \frac{\alpha}{\alpha + \beta}$$

$$\sigma^2(X) = \frac{\alpha\beta}{(\alpha + \beta)^2(\alpha + \beta + 1)}.$$

As for the cumulative distribution function, CDF, it is expressed as follows:

$$F(x, \alpha, \beta) = \frac{\gamma(x, \alpha, \beta)}{\Gamma(\alpha, \beta)}$$

Here, $\gamma(x, \alpha, \beta)$ is the incomplete Beta function. Figure 13.10 depicts the pdf and CDF of the Beta distribution.

If we have n observations (x_1, \dots, x_n) and \bar{x} and σ are the sample mean and sample standard deviation, respectively, then we can write:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^N x_i, \sigma^2 = \frac{1}{n} \sum_{i=1}^N (x_i - \bar{x})^2.$$

The moments scheme approximates the parameters as given below:

$$\alpha = \bar{x} \left(\frac{\bar{x}(1 - \bar{x})}{\sigma^2} - 1 \right)$$

$$\beta = (1 - \bar{x}) \left(\frac{\bar{x}(1 - \bar{x})}{\sigma^2} - 1 \right).$$

Some simplifications have been devised to the Beta distribution [41]. Examples include two random variables as given pdf functions, g_1 and g_2 shown below:

- $g_1(x, a, b, p, q) = \frac{|a|}{b^{ap} B(p, q)} x^{ap-1} \left(1 - \left(\frac{x}{b}\right)^a\right)^{a-1}$, for $0 < x^a < b^a, a > 0, p > 0, q > 0$.
- $g_2(x, a, b, p, q) = \frac{|a| x^{ap-1}}{b^{ap} B(p, q) \left(1 + \left(\frac{x}{b}\right)^a\right)^{p-q}}$, for $0 < x < \infty, a > 0, p > 0, q > 0$.

The first type of pdf, g_1 , includes Pareto and Gamma and few other distributions. For instance, Pareto distribution is inferred by $\text{Pareto}(x, b, p) = g_1(x, -1, b, p, -1)$ and the Gamma distribution is deduced from $\lim_{a \rightarrow 0} g$. Furthermore, the second category, g_2 , fits several essential distributions as unique cases including the exponential, Weibull, gamma, log-normal, and others.

It was found that the Beta distribution and its generalized distributions have been proven to be extremely beneficial in the study of family revenue, daily stock revenue, and the approximation of the slope of regression paradigms. Moreover, it is possible to use it as a rough model in the lack of data distribution of a random quantity such as the quantity of defective items in a sample, portion of packets that need to be retransmitted due to error, and fragment of remote procedure calls (RPCs) taking more than a specific amount of time, among others.

13.4.14 Binomial Distribution

Binomial distribution abridges the number of tests, or observations when each test has the same probability of achieving one particular value. It defines the probability of experiencing an itemized number of profitable outcomes in a defined number of attempts.

The binomial distribution is the discrete distribution that is known to be associated with the problems that study number of 1's in a sequence of n independent $\{0, 1\}$ experiments, where the outcome produces success with a given probability p . Such a success/failure test is also called a Bernoulli trial. Thus, when $n = 1$, then the binomial distribution is the Bernoulli distribution. The binomial distribution simulates the whole number of successes in replicated trials from an endless population under the following scenarios: (a) All trials are independent of each other, (b) only two outcomes are achievable on each of n trials, and (c) the probability of success for each trial is steady.

Here, a random variable X that obeys the binomial distribution and has parameters n and p will have a probability of obtaining just k successes specified by the probability mass function (pmf), f , expressed by:

$$f(k, n, p) = \binom{n}{k} p^k (1 - p)^{n-k} \text{ for all } k = 0, 1, 2, \dots, n$$

Here, $\binom{n}{k}$ is the binomial coefficient is read as “ n choose k ,” which can also be written as $C(n, k)$, ${}_n C_k$, or ${}^n C_k$.

Since k successes can happen anywhere in a succession of n events, if $k \leq n$, the probability of k successes knowing their location is specified by $p^k (1 - p)^{n-k}$. The expression obeys the fact that the number of likely locations of k successes within a succession of n events is given by: $\binom{n}{k}$.

Since the probability of calculating the probability of $n - k$ successes, knowing that the likelihood of a success is equal to $1 - p$, is basically the probability of having n successes; knowing that the probability of a success is equal to p , one can infer that the probability mass function (pmf), $f(-)$, satisfies the following expression:

$$f(k, n, p) = f(n - k, n, 1 - p)$$

The cumulative distribution function, CDF, can be expressed as follows:

$$F(k, n, p) = P(X \leq k) = \sum_{j=0}^k \binom{n}{j} p^j (1 - p)^{n-j},$$

stipulated that k is an integer and $0 \leq k \leq n$. (Fig. 13.11)

If X is a binomial distributed random variable (represented by $\approx B(n, p)$), then the expected value and standard deviation of X can be expressed by:

$$\begin{aligned} E(X) &= \sum_{k=0}^n kP(X = k) = \sum_{k=0}^n k \binom{n}{k} p^k (1 - p)^{n-k} \\ &= \sum_{k=1}^n np \binom{n-1}{k-1} p^{k-1} (1 - p)^{n-1-(k-1)} = np \sum_{j=0}^{n-1} \binom{n-1}{j} p^j (1 - p)^{n-1-j} = np \\ \sigma^2(X) &= E((X - np)^2) = np(1 - p). \end{aligned}$$

Here, the variance can be calculated directly or by exploiting the following explanations:

For $n = 1$, it can be simply established that:

$$\sigma_1^2 = (1 - p)^2 p + (0 - p)^2 (1 - p) = p(1 - p);$$

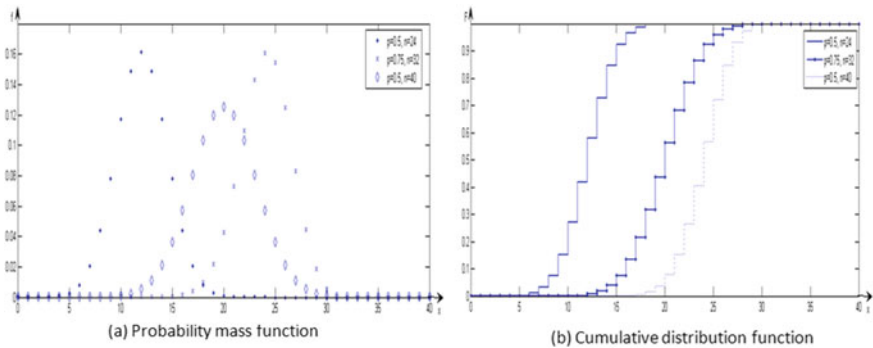


Fig. 13.11 Binomial probability mass function (pmf) and cumulative distribution function

Now assume that n is general, so given that the trials are independent, we can add the variances for each trial, supposing that X is the sum of n independent Bernoulli variables. Thus, the expression below holds:

$$\sigma_1^2 = \sum_{k=1}^n \sigma_1^2 = np(1-p).$$

Parameter approximation is the process of finding the parameter, p , of the binomial distribution that matches well an agreed-on set of trials of a binomial distributed random variable X . Thus, if we have n observations (x_1, \dots, x_n) , then p can be estimated by:

$$\hat{p} = \frac{1}{n} \sum_{i=1}^N x_i.$$

13.4.15 Negative Binomial Distribution

The negative binomial distribution represents the number of successes beforehand a given number of failures are attained in an independent series of recurrent indistinguishable trials. Its parameters are the likelihood of success in a single trial p and the quantity of failures, represented by r .

A unique case of the negative binomial distribution is when $r = 1$, which gives a geometric distribution that models the number of successes before the first failure as explained earlier. Usually, parameter r can have non-integer quantities. The negative binomial is suitable in modeling tally of data. If the r parameter is an integer, then the pdf can be expressed as follows:

$$f(k, r, p) = \binom{r+k-1}{k} p^r (1-p)^k$$

Here, r is not an integer. The binomial coefficient in the delineation of the probability mass function, pmf, is substituted by a corresponding expression using the function Γ , which can be expressed by:

$$f(k, r, p) = \frac{\Gamma(r+k)}{\Gamma(r)\Gamma(k+1)} p^r (1-p)^k$$

13.4.16 Chi-Square Distribution

The chi-square distribution (also referred to as χ^2 distribution) is one of the most widely used distributions in statistical significance test [32, 42]. It is the distribution of the summation of squared standard normal deviates. The degrees of freedom of

the distribution are equivalent to the number of standard normal deviates being summed; hence, chi-square with one degree of freedom, $\chi^2(1)$, is basically the distribution of a single normal deviate squared.

Chi-square distribution is useful since under realistic assumptions, easily calculated quantities can be established to have distributions that can be estimated by χ^2 distribution if the null hypothesis is correct [32, 43, 44]. The χ^2 distribution has one parameter, designated by k , which is a positive integer, which denotes the number of degrees of freedom as shown below.

Consider k random variable $X_i, i \leq k$, that are independent and normally distributed with mean value 0 and standard deviation 1, then the random variable X expressed by:

$$X = \sum_{k=1}^n X_k$$

The χ^2 distribution is a distinctive case of the gamma distribution. The probability density function (pdf) of the χ^2 distribution can be expressed by:

$$f(x, k) = \begin{cases} \frac{1}{2^{k/2}\Gamma(k/2)} x^{-1+k/2} e^{-x/2}, & x \geq 0 \\ 0, & x < 0 \end{cases}$$

Here, Γ represents the gamma function.

The cumulative distribution function, CDF, of the χ^2 distribution X is given by

$$\begin{aligned} F(x, k) &= P(X \leq x) = \int_0^x \frac{1}{2^{k/2}\Gamma(k/2)} t^{-1+k/2} e^{-t/2} dt \\ &= P(X \leq x) = \frac{\gamma(k/2, x/2)}{\Gamma(k/2)} \end{aligned}$$

Here, $\gamma(-, -)$ is the lower incomplete gamma function. Furthermore, the expected value of a random variable having chi-square distribution with k degrees of freedom is k and the standard deviation is $\sqrt{2k}$.

If X is a χ^2 distributed random variable of k degrees of freedom, then as k goes to infinity, the distribution of X turns to normal distribution. Moreover, it has been found that $\sqrt{2X}$ is nearly normally distributed with a mean equals to $\sqrt{2k - 1}$ and a standard deviation equals to 1.

There are special tables of the χ^2 distribution, which are commonly available, and the function is integrated in numerous spread sheets and statistical tools.

The chi-square distribution was first originated by Karl Pearson in 1900, and in his original article, he signified the sum by the character χ^2 and since then statisticians and modelers have referred to this distribution as chi-square (χ^2) distribution. In general, χ^2 distribution is used when a sum of squares of normal variables is

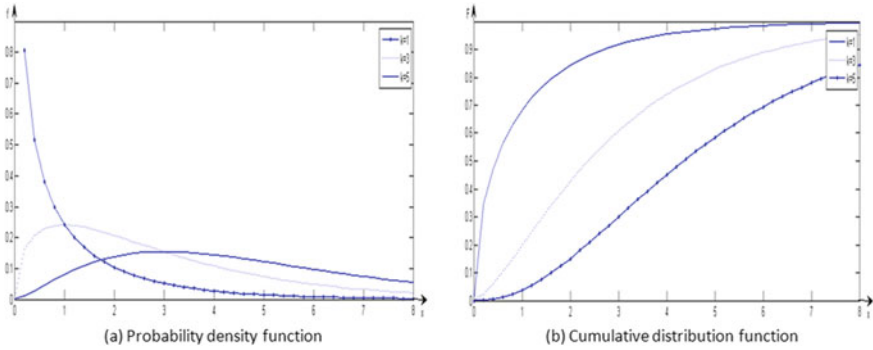


Fig. 13.12 Chi-square probability density function and cumulative distribution function

affected, as for modeling sample variances. Figure 13.12 shows the pdf and CDF of the chi-square distribution.

Other related distributions to chi-square distribution include the inverse chi-square distribution and chi distribution.

13.4.17 F-Distribution

The *F*-distribution is a skewed distribution of probabilities like chi-squared distribution; however, the chi-squared distribution deals with the degree of freedom with one group of variables, but the *F*-distribution deals with several levels of events with various degrees of freedom. Thus, there are a number of forms of the *F*-distribution for different levels of degrees of freedom. *F*-distribution is considered as a continuous probability distribution, and sometimes, it is called the Fisher–Snedecor distribution. It is described using two χ^2 distributed variables, X_1 and X_2 , as follows. This, if X is a random variable defined as shown below:

$$X = \frac{k_2 X_1}{k_1 X_2}$$

then X has an *F*-distribution if X_1 or X_2 has a χ^2 distribution with k_1 or k_2 parameters, respectively. Furthermore, X_1 and X_2 must be independent. Typically, the *F*-distribution arises often as the null distribution of a test statistic, and most particularly in the analysis studies of variance.

The probability density function (pdf) of an *F* distributed random variable $X(X \approx F(k_1, k_2))$ is expressed as follows:

$$f(x, k_1, k_2) = \frac{1}{xB(k_1/2, k_2/2)} \left(\frac{k_1 x}{k_1 x + k_2} \right)^{k_1/2} \left(\frac{k_2}{k_1 x + k_2} \right)^{k_2/2}$$

Here, x is a non-negative real value, d_1 and d_2 are the degrees of freedom, and B is the Beta function.

F-distribution is utilized in hypothesis testing and in modeling the ratio of sample variates.

13.4.18 Student's t-Distribution

The Student's t -distribution, (also called t -distribution) is a probability distribution that is applied to estimate population parameters while sample size is small and/or when the population variance is undetermined.

Based on the central limit theorem, the sampling distribution of a statistic such as an expected value of a sample will obey a normal distribution, providing the sample size is appropriately large. Given this, if we identify the standard deviation of the population, we can determine a z -score and employ the normal distribution to assess probabilities with the sample mean. Nevertheless, sample sizes are at times small, and frequently, we do not have the standard deviation of the population. If either of these issues happens, modelers depend on the distribution of the t statistic or sometimes called the t score.

The t -distribution is a probability distribution that is used in the problem of estimating the mean of a normally distributed population when the sample size is small. It was realized by William V. Gosset, [43–45], through his work at Guinness brewery. It is a family of curves depending on a single parameter (the degrees of freedom). As the degree of freedom $\hat{T}^1/2$ shots to infinity, the t -distribution congregates to the standard normal distribution.

Assume that we have n independent random variables X_1, \dots, X_n which are normally distributed with mean μ and standard deviation σ and \bar{X}_n and S_n are their sample mean and sample standard deviation, then we can express the mean and standard deviation as follow:

$$\bar{X}_n = \frac{1}{n} \sum_{k=1}^n X_k, \quad S_n = \frac{1}{n-1} \sum_{k=1}^n (X_k - \bar{X}_n)^2$$

It can be easily demonstrated that the variable Z expressed by:

$$Z = \frac{\sqrt{n}}{\sigma} (\bar{X}_n - \mu)$$

is normally distributed with mean 0 and variance 1, as the sample mean \bar{X}_n is normally distributed with mean μ and standard deviation $\frac{\sigma}{\sqrt{n}}$. The T variable is expressed by [32, 43]:

$$T = \frac{\sqrt{n}}{S_n} (\bar{X}_n - \mu)$$

And is shown to have the following probability density function, pdf:

$$f(x) = \frac{\Gamma(n/2)}{\sqrt{(n-1)\pi}\Gamma((n-1)/2)} (1 + x^2/(n-1))^{-n/2}.$$

The parameter $(n-1)$ is called the number of degrees of freedom. It can be seen that the distribution depends only on v and not μ or σ . Such an attribute makes the t -distribution very special in theory and practice. The t -distribution is associated with the F-distribution as follows. The square of a t -distributed random variable with $n-1$ degrees of freedom is an F-distribution with $n-1$ degrees of freedom. Assume that the number α is chosen in a way that:

$$P(-\alpha < T < \alpha) = \beta$$

where T is t -distributed with $n-1$ degrees of freedom. The inequality above is equivalent to:

$$P\left(\bar{X}_n - \frac{S_n}{\sqrt{n}}\alpha < \mu < \bar{X}_n + \frac{S_n}{\sqrt{n}}\alpha\right) = \beta$$

Thus, we can say that the interval $\left[\bar{X}_n - \frac{S_n}{\sqrt{n}}\alpha, \bar{X}_n + \frac{S_n}{\sqrt{n}}\alpha\right]$ is a β -percent confidence interval for μ . There are particular handy tables available for t -distribution for various degrees of freedom values [32, 43].

13.4.19 Concluding Remarks

To conclude, we have reviewed the key features of the frequently used distributions in modeling and simulation. These distributions can be continuous while the others can be discrete. Among the major popular probability distributions are the uniform, exponential, normal, Poisson distribution, Weibull, Pareto, Geometric, Beta, binomial, Gamma, Erlang, chi-square, chi-distribution, inverse chi-distribution, F-distribution, and Student's t -distribution. We reviewed the main characteristics of these distributions and their key applications.

13.5 Queuing Theory

Claudia Szabo

Queueing theory, first advanced by Agner Erlang [46], refers to the mathematical study of the formation and function of queues. The two key elements of queueing systems are customers and servers, where customer refers to anything that requests or needs a service, and a server is anything that provides that service. There are many examples of queueing systems, including reception desks, airplanes, nurses, computing servers, and web servers among others.

There are several characteristics of a queuing system, including the calling population (the population of all potential customers), system capacity (the limit to the number of customers that may be waiting in the line), arrival process (describing how frequently customers arrive in the system), queuing behavior (how customers are served), and the service time and mechanism (how long does it take for requests to be served, and how they are served: first in first out—FIFO; last in first out—LIFO, etc.) [47].

Recognizing the many different types of queueing systems and the variety of customer and server-specific characteristics, [48] proposed a notation for parallel server systems based on the format $A/B/c/N/K$, where:

- A —is the inter-arrival time distribution (common are M —exponential, Markov; D —deterministic, G —general)
- B —is the service time distribution (common are M —exponential, Markov; D —deterministic, G —general)
- c —is the number of servers
- N —is the system capacity
- K —is the size of the calling population

The most well-known example is the single-server queue, $M/M/1/\infty/\infty$, which is usually shortened to $M/M/1$.

As queueing systems are ubiquitous, understanding their measure of performance is critical. Several measures of performance include server utilization, system throughput, the time average number of customers in the systems (L), and the average time spent per customer (w). The conservation law or Little's Law $L = \lambda w$ is valid for any queueing system, where λ captures the customers' inter-arrival rate [49].

13.6 Statistics

Paul Weirich

A statistical test of a hypothesis collects data bearing on the hypothesis. For example, the hypothesis may be about the mean value of a trait of members of a population, and the test may obtain the mean value of the trait in a random sample of the population.

Statistical methods fall into three camps. Frequentist methods do not attribute a probability to a hypothesis or to possible results of a statistical test of a hypothesis. They classify a test result's type as probable or improbable given the hypothesis, and they do this using probability in the frequentist sense, so that, more precisely, they use the relative frequency of the test result's type in a long run of repetitions of the statistical test, assuming the hypothesis's truth.

Likelihoodism holds that the import of the evidence that a statistical test provides concerning a hypothesis and its rivals is completely expressed by the likelihoods of the hypotheses on the evidence, that is, the probability of the evidence given each of the hypotheses. It does not attribute probabilities to the hypothesis and its rivals, or to possible results of the statistical test.

In contrast, Bayesian methods, applied in the context of a statistical test of a hypothesis, attribute probabilities to the hypothesis and to the possible results of the test. They use Bayes' theorem to obtain the probability of the hypothesis given the evidence, and they use a type of probability that applies to events, rather than to event types, and is relative to evidence, including background information.

Selection of a statistical method from the three camps remains controversial, as each method seems to have failings as well as appealing features. For example, frequentist methods fail to provide a probability of a hypothesis given the result of a statistical test of the hypothesis, that is, a probability to guide practical action when consequences depend on the hypothesis's truth-value. Bayesian methods use, but do not offer an objective way of obtaining, the probability of a hypothesis prior to a statistical test of the hypothesis and the probabilities of the possible results of the statistical test. For additional in-depth information on these topics, the reader is referred to publications [50–56].

References

1. Wikipedia (2020, November) Systems science. https://en.wikipedia.org/wiki/Systems_science
2. Mobus GE, Kalton MC (2015) Principles of systems science. Springer, New York
3. von Bertalanffy L (1969) General systems theory—foundations, development, Applications. George Braziller, Inc., New York, NY
4. Ashby WR (1961) An introduction to cybernetics. Chapman & Hall Ltd., London, UK
5. Mesarovic M, Takahara Y (1975) General systems theory: mathematical foundations. Academic Press, London, UK
6. Klir GJ (1985) Architecture of systems problems solving. Springer, New York
7. Mesarovic M, Takahara Y (1989) Abstract systems theory. Springer, New York, NY
8. Wymore AW (1967) A mathematical theory of systems engineering—the elements. Wiley, New York, NY
9. Kalman RE, Falb PL, Arbib MA (1969) Theories of abstract automata. Prentice-Hall, Englewood Cliffs, NJ
10. Wymore AW (1993) Model-based systems engineering. CRC Press LLC, Boca Raton, FL
11. INCOSE (2014) INCOSE system engineering vision 2025. <https://www.incose.org/docs/default-source/aboutse/se-vision2025.pdf?sfvrsn=4&sfvrsn=4>. Accessed July 2014
12. INCOSE (2014) INCOSE future of systems engineering (FuSE). Available <https://www.incose.org/about-systems-engineering/fuse>. Accessed November 2020
13. Zeigler BP (1984) Multifaceted modelling and discrete event simulation. Academic Press, London
14. Wach P, Salado A (2020) A systematic literature review on the math of model-based systems engineering (Tech. Report, VATEch)
15. Zeigler BP, Marvin J, Cadigan JJ (2018) Systems engineering and simulation: converging toward noble causes. In: Winter simulation conference, Gothenburg, Sweden
16. Gear CW (1971) Numerical initial value problems in ordinary differential equations. Prentice Hall PTR, Upper Saddle River, NJ, USA

17. Zeigler BP, Muzy A, Kofman E (2019) *Theory of modeling and simulation*, 3rd edn. Academic Press
18. Giambiasi N, Carmona J-C (2006) Generalized discrete event abstraction of continuous systems: GDEVs formalism. In: *Simulation modelling practice and theory*. Elsevier, vol 14, pp 47–70
19. Zeigler BP, Lee J (1998) Theory of quantized systems: formal basis for DEVS/HLA distributed simulation environment. In: *Proceedings of SPIE*, pp 49–58
20. Kofman E, Junco S (2001) Quantized-state systems: a DEVS approach for continuous system simulation. *Trans Soc Comput Simul Int* 18(3):123–132
21. Kofman E (2003) Quantization-based simulation of differential algebraic equation systems. *Trans Soc Comput Simul Int* 79(7):363–376
22. D'Abreu MC, Wainer GA (2003) Hybrid dynamic systems: models for continuous and hybrid system simulation. In: *WSC '03: Proceedings of the 35th conference on winter simulation*, Winter Simulation Conference, pp 641–649
23. Cellier FE, Kofman E (2006) *Continuous system simulation*. Springer, Berlin, Heidelberg
24. Capocchi L, Federici D, Yazidi A, Henao H, Capolino G (2009) New trends and solutions for the simulation of electrical circuits using a discrete event approach. In: *2009 35th annual conference of IEEE industrial electronics*, Porto, pp 4365–4370. <https://doi.org/10.1109/IECON.2009.5414910>
25. Migoni G, Kofman E, Cellier F (2012) Quantization-based new integration methods for stiff ordinary differential equations. *Simulation* 88(4):387–407. <https://doi.org/10.1177/0037549711403645>
26. Fernández J, Kofman E, Bergero F (2017) A parallel quantized state system solver for ODEs. *J Parallel Distrib Comput* 106:14–30
27. Billingsley P (2012) *Probability and measure*, Anniversary. Wiley, Hoboken, NJ
28. Durrett R (2019) *Probability: theory and examples*, 5th edn. Cambridge University Press, Cambridge
29. Hacking I (2001) *Probability and inductive logic*. Cambridge University Press, Cambridge
30. Hájek A (2019) Interpretations of probability. In: Zalta E (ed) *The stanford encyclopedia of philosophy*. <https://plato.stanford.edu/archives/fall2019/entries/probability-interpret/>
31. Tijms H (2007) *Understanding probability: chance rules in everyday life*. Cambridge University Press, Cambridge
32. Obaidat MS, Boudriga N (2010) *Fundamentals of performance evaluation of computer and telecommunications systems*. Wiley
33. Obaidat MS, Nicopolitidis P, Zarai F (eds) *Modeling and simulation of computer networks and systems: methodologies and applications*. Elsevier
34. Balakrishnan N, Basu AP (1996) *The exponential distribution: theory, methods, and applications*. Gordon and Breach, New York
35. Ahrens JH, Deiter U (1974) Computer methods for sampling from gamma, beta, poisson and binomial distributions. *Computing* 12:223–246
36. Bryc W (2005) *Normal distribution: characterizations with applications*. Lecture notes in statistics, vol 100, 1995 (Revised June 7, 2005)
37. Abramowitz M, Stegun IA (eds) *Handbook of mathematical functions with formulas, graphs, and mathematical tables*. Dover, New York
38. Weibull W (1951) A statistical distribution function of wide applicability. *J Appl Mech* 18:293–297
39. Erlang AK (1917) Solution of some problems in the theory of probabilities of significance in automatic telephone exchanges. *Post Office Electr Eng J* 10:189–197
40. Evans M, Hastings N, Peacock B (2000) *Statistical distributions*. In: *Beta distribution*, 3rd edn. Wiley, New York, pp 34–42
41. McDonald JB, Xu YJ (1995) A generalization of the beta distribution with applications. *J Econometrics* 66(1):133–152

42. Wilson EB, Hilferty MM (1931) The distribution of Chi-Square. In: Proceedings of the National Academy of Sciences, vol 17, pp 688–689
43. Gosset WS (1908) The probable error of a mean. *Biometrika* 6(1):125
44. Obaidat MS, Papadimitriou GI (2003) Applied system simulation: methodologies and applications. Springer
45. Papadimitriou GI, Sadoun B, Papazoglou C (2003) Fundamentals of system simulation. In: Obaidat MS, Papadimitriou GI (eds) Applied system simulation: methodologies and applications. Springer, pp 9–40
46. Erlang AK (1909) The theory of probabilities and telephone conversations. *Nyt Tidsskr Matematik B* 20:33–39
47. Banks J (2005) Discrete event system simulation. Pearson Education India
48. Kendall MG, Hill AB (1953) The analysis of economic time-series-part i: prices. *J Roy Stat Soc Ser A (Gen)* 116(1):11–34
49. Little JDC (1961) A proof for the queuing formula: $L = \lambda W$. *Oper Res* 9(3):383–387. <https://doi.org/10.1287/opre.9.3.383>. <https://doi.org/10.1287/opre.9.3.383>. [JSTOR167570](https://doi.org/10.1287/opre.9.3.383)
50. Turkman MAA, Daniel CD, Müller P (2019) Computational Bayesian statistics. Cambridge University Press, Cambridge
51. Greco D (2011) Significance testing in theory and practice. *Br J Philos Sci* 62:607–637
52. Howson C, Urbach P (2006) Scientific reasoning: the Bayesian approach. In: Classical inference: significance tests and estimation, Chap. 5, 3rd edn. Open Court, Chicago
53. Sober E (2008) Evidence and evolution. In: Evidence, Chap. 1. Cambridge University Press, Cambridge
54. Utts J (2013) Seeing through statistics, 4th edn. Cengage Learning, Stamford, CT
55. Walley P (1991) Statistical reasoning with imprecise probabilities. Chapman and Hall, London
56. Young GA, Smith RL (2010) Essentials of statistical inference. Cambridge University Press, Cambridge



Supporting Engineering Areas

14

Andrea D'Ambrogio , Chen Yang, and Hessam S. Sarjoughian

Abstract

The engineering disciplines supporting, and supported by, simulation are in the scope of this chapter of the SCS M&S Body of Knowledge. The chapter provides descriptions for systems engineering, virtual and augmented reality engineering, and visualization engineering.

Keywords

Modeling and simulation · Systems engineering · M&S based systems engineering · Simulation systems engineering · Life cycle engineering · Virtual reality · Augmented reality · Human-Computer Interaction Technology (HCI)

14.1 Systems Engineering

Andrea D'Ambrogio

Systems engineering is “a transdisciplinary approach and means to enable the realization of successful systems, which must satisfy the needs of their customers, users, and other stakeholders”, as reported in the Guide to the Systems Engineering

A. D'Ambrogio (✉)
University of Roma Tor Vergata, Rome, Italy
e-mail: dambro@uniroma2.it

C. Yang
Beijing Institute of Technology, Beijing, China

H. S. Sarjoughian
Arizona State University, Tempe, AZ, USA

Body of Knowledge (SEBoK), an effort that aims at sharing and centralizing experiences and successful practices, currently governed by the International Council on Systems Engineering (INCOSE), the IEEE Systems Council and the Systems Engineering Research Center (SERC) [1]. Notwithstanding the several success stories resulting from the application of systems engineering processes, many projects still experience time and cost overruns or fail to deliver systems that meet the needs of customers, users, and other stakeholders.

Systems development processes have traditionally focused on document-centric approaches based on the use of documents and data available at different levels of abstraction and using different notations. A significant step forward has been achieved with the introduction of *Model-Based Systems Engineering (MBSE)*, which refers to the formalized application of modeling to support the system development, along all the different development phases [2].

With MBSE, a common model of the system architecture is used to support and drive the engineering process [3], so to bring significant advantages over the document-centric approach in terms of improved quality, enhanced communication and stakeholder engagement, increased productivity, enhanced knowledge transfer, and reduced risks.

Modeling and simulation (M&S)-based systems engineering further promotes executable models and proposes simulation as the native mechanism to address measures of performance and effectiveness throughout conceptual design, development, and later systems life cycle phases, so to overcome the previously mentioned difficulties in terms of budget overruns and systems quality [4].

As the system complexity increases, the executable system architecture (i.e., the simulation model) may become so complex that it is necessary to assess it not only as a valid support of SE processes but also as an objective of SE efforts. Executing an interdisciplinary systems engineering process for developing, maintaining, and employing simulations, which enable systems engineers to experiment and gain insights about the systems of interest, is referred to as *simulation systems engineering* (Durak and [5]).

The following subsections provide more details about M&S-based systems engineering and simulation systems engineering, respectively, while Sect. 14.1.3 compares the reference life cycle processes practically adopted in the M&S and systems engineering fields and proposes an integrative approach that allows systems engineers and M&S practitioners to fully catch the benefits of using M&S for systems engineering and vice versa.

14.1.1 M&S-Based Systems Engineering

The development of complex systems strongly benefits from the adoption of quantitative analysis techniques that enable a prompt evaluation of the system behavior, so to assess, before starting implementation or maintenance activities, whether or not the to-be system is going to satisfy the stakeholder requirements and constraints. In this context, M&S-based approaches may be effectively introduced

to enact evaluation of various structural and/or behavioral properties of the system under study, both at design-time and at run-time.

The adoption of M&S-based approaches is widely recognized as a cost-effective alternative to the development of experimental prototypes and as a valuable strategy to mitigate the risk of time/cost overrun due to redesign and re-engineering activities [4], yet their potential and organized implementation in the systems engineering field are not yet fully exploited, even though there are plenty of applications in specialty engineering fields, as well as in other fields, such as physical science, social science, computing, medical research, military, business, and finance.

In order to enact a successful synergy between M&S and systems engineering disciplines, significant effort has been spent at educational, theoretical, methodological, and professional levels.

In this respect, at *educational* level, the book from Loper provides an introduction to the fundamental concepts of M&S and systems engineering, and how M&S is used in the systems engineering life cycle [6]. The book can be viewed as a handbook that introduces the reader to the broad discipline of M&S that systems engineers need to understand to be effective in their jobs.

The *theoretical and methodological* contributions of M&S to systems engineering are clearly explained in focused editorial efforts. The book from [4] collects reusable M&S lessons for systems engineering, offering an initial mean for cross-domain capitalization of the knowledge, methodologies, and technologies developed in several communities, and identifying a way forward to use the potentials of M&S for better systems engineering in the domains of architecture alignment and improvement, collaboration tools and repositories, and theoretical foundations [4].

An additional useful resource is the book from [7], which represent an important and decisive step forward in demonstrating the theoretical and methodological contributions of M&S to system-of-systems (SoS) engineering, through the lenses of application. The book addresses how M&S approaches assist system engineers to appropriately face critical challenges of SoS, which are required to integrate and coordinate multiple systems so to achieve levels of performance and offer capabilities that are beyond the grasp of individual constituent systems. The use of M&S to properly address the scalability, adaptiveness, self-organizing, interoperability, independence, and emergent behavior properties of SoS is also illustrated in the book from [8], which introduces a unified process based on the well-known discrete event systems specification (DEVS) formalism and a formal computational complex dynamical systems framework. In addition to M&S approaches based on discrete-event model representation, the use of agent-based simulation approaches to support systems engineering is clearly and extensively addressed in the book from [9]. The theoretical considerations and the tools required to enable the study of emergent behaviors in complex systems engineering are addressed in the book from [10], which offers a number of M&S-based methods, technologies, and approaches that are designed to broaden the understanding of emergent behavior in complex systems. A recent addition to the aforementioned resources is the book from [11], which provides a compendium of the state of the art in cloud-based M&S as a

medium for facilitating engineering and governance of cyber-physical systems, the next generation of systems for a highly connected society.

Finally, at *professional* level, a successful implementation of M&S-based systems engineering requires an effective integration of M&S and systems engineering life cycles. In this respect, Sect. 14.1.3 illustrates an attempt to integrate the life cycle processes currently used by systems engineers and M&S practitioners, by specifically addressing the ISO 15288 (Systems and Software Engineering—System Life Cycle Processes) standard and the IEEE 1730-2010 [12] (Recommended Practice for Distributed Simulation Engineering and Execution Process—DSEEP) standard, respectively.

14.1.2 Simulation Systems Engineering

The inherently compositional and distributed nature of complex systems make the use of distributed simulation approaches a natural fit. A *distributed simulation (DS)* results from the orchestration of a number of simulation components that essentially mirror the component-based structure of the system under study.

However, the implementation of a DS is a challenging task in terms of the required effort and the significant know-how that is needed to properly use the available frameworks, such as the *high-level architecture (HLA)*, and the related implementation technologies (IEEE, [13]). In some cases, the development of a DS is seen as a task of complexity comparable to the development of the system under study [14].

For such reasons, the DS implementation activities are to be carried out according to a well-defined engineering process that effectively supports the DS development through the use of formal models and the introduction of a significant degree of automation. This is obtained by using advanced approaches that apply metamodelling techniques and *automated model transformations*, introduced in the more general model-driven engineering (MDE) context, to increase the level of automation throughout the DS life cycle, thus reducing the costs of DS development and maintenance activities.

Model-driven engineering (MDE) is an approach to system design and implementation that addresses the raising complexity of execution platforms by focusing on the use of formal models [15]. According to this paradigm, the abstract models that are specified at the beginning of the system life cycle are then given as input to model transformations that generate models at lower levels of abstraction, until stepwise refined models can be made executable.

One of the most important initiatives driven by MDE is the *model driven architecture (MDA)*, the object management group (OMG) incarnation of MDE principles [16]. MDA introduces standards for specifying technology neutral metamodels (i.e., models used to describe other models), for specifying model-to-model and model-to-text transformations, and for serializing MOF metamodels/models into XML-based schemas/documents.

The core idea of MDA in the DS domain is base DS engineering on a series of model transformations that replace some of the manual effort applied by DS experts with precise transformation rules and formal evaluations.

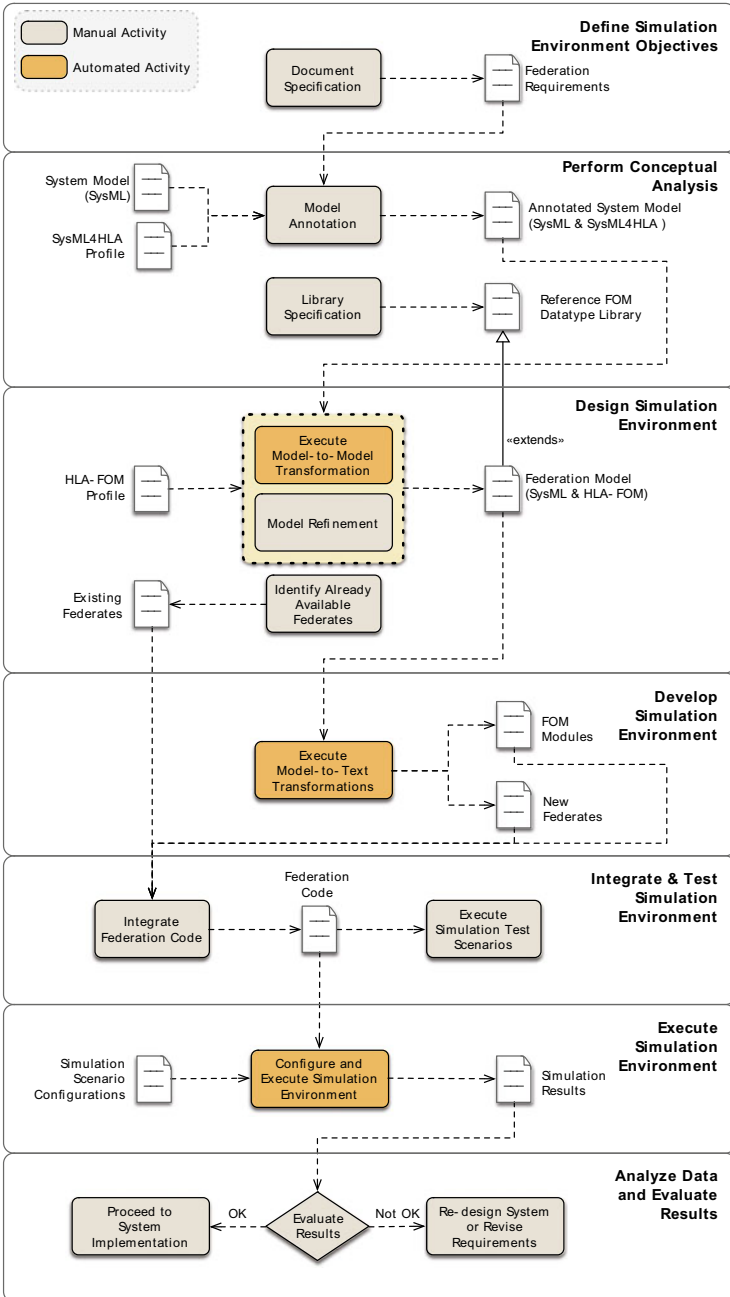


Fig. 14.1 Model-driven DSEEP enhancement [17]

Figure 14.1 illustrates a proposed enhancement of DSEEP that benefits from the adoption of model-driven engineering principles and that has been tailored to fit the systems engineering domain's needs [17].

The relevance of model-driven approaches for researchers and practitioners in the DS field is witnessed by the various editions of the International Workshop on Model-Driven Simulation Engineering (Mod4Sim), started in 2011 by D'Ambrogio and [18] within the SCS Spring Simulation multi-conference, by the book from [19], and by tutorials offered in the context of M&S-focused conferences (see, e.g., [17]). Model-driven approaches start being applied to carry out DS efforts at industrial level also, such as in the "HRAF: EDLS Distributed Simulation Federation and Model-driven Engineering Framework Development" project, funded by the European Space Agency [20].

14.1.3 Bridging the Gap Between M&S and Systems Life Cycle Processes

In order to help realize successful systems, the systems engineer supports a set of life cycle processes beginning early in conceptual design and continuing throughout the life cycle of the system through its manufacture, deployment, use, and disposal. The systems engineer must analyze, specify, design, and verify the system to ensure that its functional, interface, performance, physical, and other quality characteristics, and cost are balanced to meet the needs of the system stakeholders.

There are a number of different systems engineering process models available to support execution of the systems engineering efforts. A notable example is the *ISO/IEC/IEEE 15288 standard*, which results from a coordinated effort by IEEE and ISO/IEC and provides a common process framework for describing the life cycle of systems created by adopting a systems engineering approach [21]. The ISO/IEC/IEEE 15288 standard introduces a set of 30 processes, grouped into *agreement processes*, which specify the requirements for the establishment of agreements with organizational entities, *organizational project-enabling processes*, which help ensure the organization's capability to acquire and supply products or services through the initiation, support, and control of projects, *technical management processes*, which are used to establish, execute, control, and evolve management plans, and *technical processes*, which are executed to carry out the whole set of technical activities throughout the system life cycle.

A successful implementation of M&S-based systems engineering requires an integration of systems engineering and M&S life cycles, as standardized by ISO/IEC/IEEE 15288 and IEEE DSEEP, respectively.

There are several contributions that address the need to look at systems engineering and M&S as mutually supportive disciplines, as well as the need to look at simulation support for systems engineering or to apply the systems engineering process for simulation, as described in the previous sections. However, the ISO/IEC/IEEE 15288 standard does not specifically address how to exploit M&S within systems engineering efforts. Indeed, the term "simulation" occurs only in a

few sections of the standard, as a possible technique (with mathematical analysis and experimentation) to analyze technical performance, system behavior, quality characteristics, technical risks, and life cycle costs, and as a possible validation method or technique (with inspection, analysis, analogy/similarity, demonstration, peer-review, testing, and certification). Analogously, the term “simulator” (i.e., simulation implementation) is only referred to as a possible validation enabling system.

Vice versa, the INCOSE Handbook on Systems Engineering [22] recognizes M&S as a powerful and effective method, whose value increases with the size, be it physical or complexity, of the system or system of systems under development, and also underlines the fact that the completed model or simulation can be considered as a system or a product in its own right (and thus to be built and maintained through a dedicated process, as specified by the IEEE DSEEP standard).

Looking at M&S simply as one of the possible methods or techniques that may be effectively used for systems analysis and/or validation does not contribute to fully exploit its potential. M&S, and particularly distributed simulation, should be considered as necessary and cost-effective enabling systems or services to augment the capabilities of the set of processes defined by the ISO/IEC/IEEE 15288 standard.

In this respect, efforts are being carried out to make an *overlay of IEEE DSEEP onto ISO/IEC/IEEE 15288*, in addition to the standard overlays that IEEE DSEEP already provides for HLA, distributed interactive simulation (DIS), and test and training enabling architecture (TENA). It is almost clear at a first glance that IEEE DSEEP basically corresponds to the technical processes of the ISO/IEC/IEEE 15288 and has references to the technical management processes, as described in the proposed mapping illustrated in Fig. 14.2 [14].

The ultimate goal of such an overlay proposal can be stated as creating an impact in the evolution of the life cycle standards toward addressing integration of the systems engineering and DS life cycle processes.

14.2 VR/AR Engineering

Chen Yang, Bo Hu Li

14.2.1 Connotation of VR/AR Engineering

Virtual reality/augmented reality (VR/AR) engineering is an engineering branch that integrates computer science and engineering (computer graphics, computer vision, digital image processing, etc.), software engineering, electronic science and engineering, control science and engineering, psychology, and VR and AR technology. Its main target is to build and improve the engineering technology of VR/AR application and support the design and implementation of VR/AR software, hardware, and systems with practical application value for relevant domains and scenarios.

We here set forth the basic concepts of VR/AR to make further discussions easy to understand.

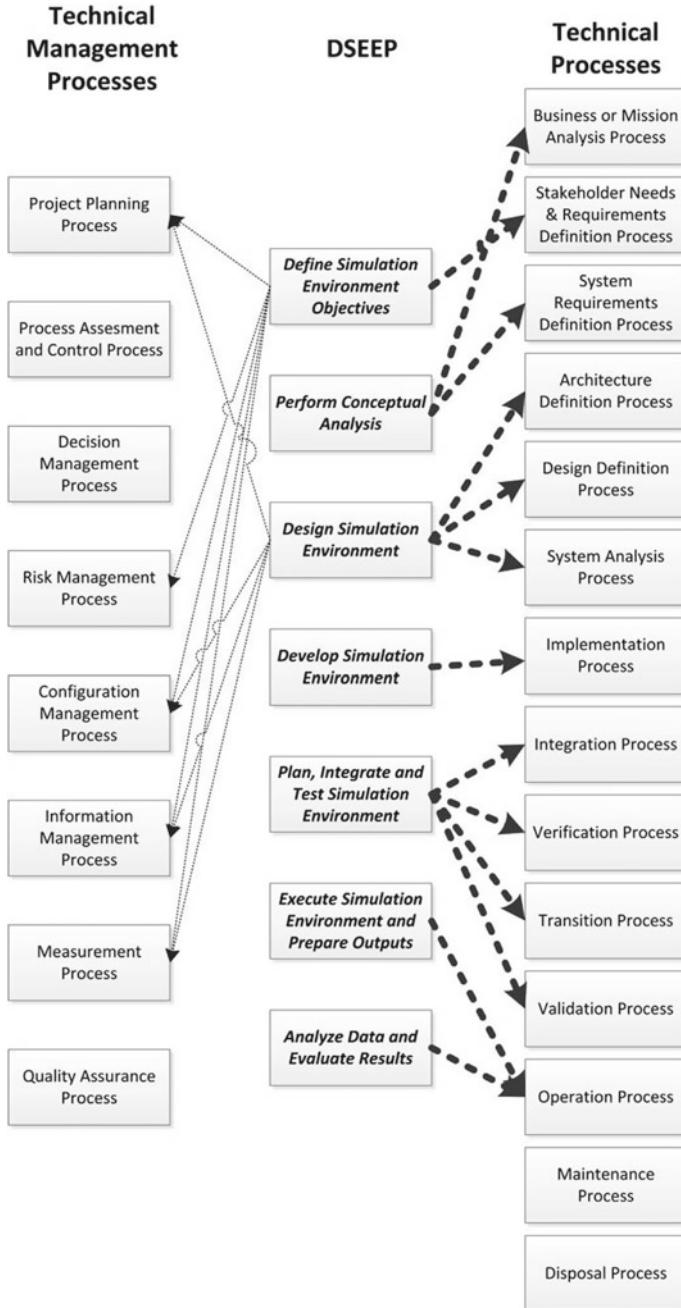


Fig. 14.2 IEEE DSEEP overlay on ISO/IEC/IEEE 15288 [14]

Virtual reality (VR for short) is the use of computer simulation to generate a three-dimensional virtual world, providing users with simulated vision, hearing, touch, and other senses so that users feel as if they were in the real environment and were able to observe things in the three-dimensional space immediately without any restrictions [23, 24]. When the user moves, the computer can immediately conduct complex calculations and transmit the accurate image of the three-dimensional world back to produce a sense of presence. This technology integrates the latest achievements of computer graphics, computer simulation, artificial intelligence, sensing, display, and parallel processing technologies. It is a high-tech simulated system generated by computer technology.

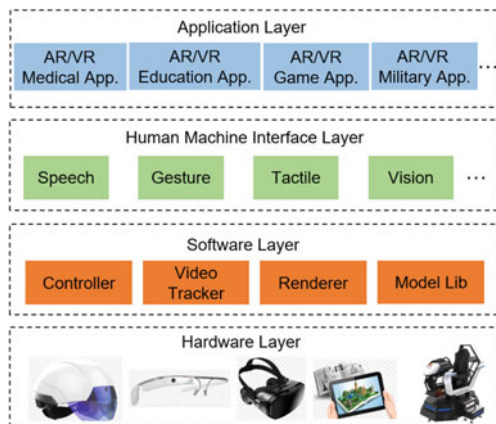
Augmented reality (AR for short) is a new technology that integrates the real-world information and the virtual world information seamlessly. Through computer technology, it simulates and superposes physical information (visual information, sound, taste, touch, etc.) that is difficult to experience in a certain spatio-temporal range of the real world. And in this way, virtual information is applied to the real world and perceived by human senses so that it delivers a sensory experience beyond reality. Real environment and virtual objects are superimposed on the same picture or in the same space in a real-time manner [25].

14.2.2 Architecture

As shown in Fig. 14.3, an architecture of the VR/AR engineering system [26] consists of four layers: hardware layer, software layer, user interface layer, and application layer, as follows:

- (1) **Hardware layer:** including VR/AR hardware infrastructure (various embedded sensors, computing systems, and display devices), such as VR/AR helmets, VR/AR smart glasses, VR/AR tablets/smartphones, VR/AR simulators, and VR/AR displays.

Fig. 14.4 Architecture for the VR/AR engineering system



- (2) **Software layer:** including controller module, video tracking module, rendering module, VR/AR model library, audio–video–image database, etc. The controller module is responsible for sensing location, viewpoint, lighting and other information, performing related computing tasks, and coordinating the video tracking module, rendering module, VR/AR model library, etc., to complete VR/AR tasks; the video tracking module is responsible for matching reference enhancement information, computing the transformation that is necessary to fit the reference enhancement information to the video image, and integrating the final enhanced information into the video graphics; the rendering module is responsible for the presentation of all the above information (current viewpoint information, enhanced information, audio and video, 3D models) to the user.
- (3) **User interface layer:** including the interaction interface between the VR/AR engineering system and users, which supports interaction modes such as voice, posture, touch, and vision.
- (4) **Application layer:** supporting VR/AR engineering applications oriented to medical care, education, games, military, and other industries.

14.2.3 Key Technologies

14.2.3.1 Human–Computer Interaction Technology

Human–computer interaction technology (HCI) is a kind of hardware and software technology that realizes the dialog between human and computers in an effective way through input and output devices of computers. The hardware technology includes the input equipment technology for calculating all kinds of information (language, expression, action, etc.) from humans and the output equipment technology for feedback information to humans from the computer; the software technology mainly includes the algorithms and technologies for the computer to accurately recognize and understand all kinds of information given by humans and give appropriate feedback. Multi-channel human–computer interaction technology is the main focus of current research efforts. It is helpful for computers to receive and process various information sent by humans and realize effective communications between men and computers on a deeper layer. Human–computer interaction technology is one of the most widely used technologies in many fields.

14.2.3.2 Haptic Feedback Technology

Haptic feedback technology is a technology that is utilized to reproduce real tactile sensation for users through a series of actions such as force or vibration provided by the professional equipment. This technology can be used to mimic the real touch in the virtual scene and bring more real experience to people. Haptic feedback technology is widely used in business, film and television, games, medical care, and other industries.

14.2.3.3 Environment and Object Modeling Technology

Environment and object modeling technology refers to the technology that can extract the characteristics or attributes of the real environment/object and build a virtual environment/object model according to the application needs.

14.2.3.4 System Integration Technology

System integration technology refers to the technology that can help integrate various technologies applied in VR/AR (such as information synchronization technology, model calibration technology, data conversion technology, recognition and synthesis technology, etc.).

14.2.3.5 3D Registration Technology

3D registration technology is one of the core technologies in reality enhancement (AR). It mainly includes the position-orientation identification and tracking technology and the localization technology of virtual objects in the real world. Real time (no delay), accuracy (no jitter of camera equipment), and robustness (not affected by lighting, occlusion, and other realistic factors) are the key elements of 3D registration technology.

14.2.3.6 3D/4D Display and Stereo Synthesis Technology

3D/4D display and stereo synthesis technology refers to a kind of technology that enables the real-time generation of stereo graphics in complex scenarios, as well as the elimination of the correlation between the direction of sound and the movement of the user's head in the virtual reality system. This technology is very important for VR/AR to reproduce the reality of virtual scenes.

14.2.3.7 Virtuality and Reality Fusion-Based Display Technologies

Virtuality and reality fusion-based display technology [27] is a technology that can support adding virtual objects to the real scene, showing the user a picture of combined virtuality and reality. It is one of the key technologies in AR, and the main display devices are generally divided into helmet display, hand-held display, projection display, etc.

14.2.4 Development Trend

14.2.4.1 Miniaturization and Mobility

Consumer demands for the portability and mobility of VR/AR devices and applications promote the development of VR/AR engineering toward miniaturization and high mobility, in order to improve user experience.

14.2.4.2 High Fidelity

The main task of VR engineering is to create a virtual world that allows users to have an immersive experience. VR engineering applications will continue to move forward in the direction of higher resolution and fidelity, delivering users a sense of immersion and real experience. Fidelity evaluation also involves VR psychology, VR sociology, and other disciplines [28].

14.2.4.3 High-Virtuality-Reality Fusion

The main target of AR engineering is to integrate virtual world information and real-world information “seamlessly” to provide users with sensory experience beyond reality, so it develops to provide more extrasensory information and realize more seamless sensory experience of virtuality-reality integration [23].

14.2.4.4 Networking

The emergence and development of new network technology (such as 5G) have greatly improved the bandwidth, real time, and reliability of the network, which will promote the development of VR/AR to the direction of networking, and facilitate the development of remote and interactive online VR/AR engineering applications with large-scale involvement of distributed users.

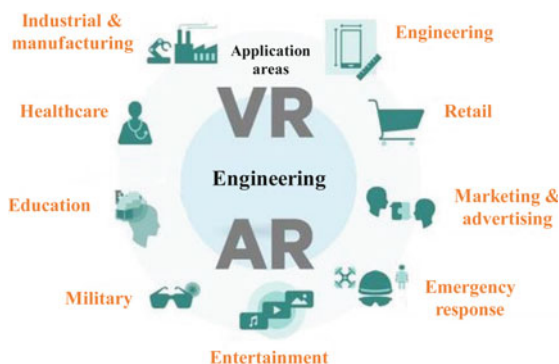
14.2.4.5 Intelligence

With the rise of research and application of big data and the Internet, learning how to efficiently build models through analyzing image, video, and domain big data has become a hotspot, and improving the adaptability of virtual environment is receiving increasing attention, making intelligence an important feature of VR/AR research and applications in the new era [23].

14.2.5 Applications

VR/AR engineering is widely used in the manufacturing industry, medicine, military, aerospace, energy, entertainment games, education, marketing, and other fields [29, 30]. Figure 14.4 shows a subset of the many application domains of VR/AR engineering. For example, in the industrial field, VR/AR can show the industrial details and operation management completely [31], in the medical field, doctors can use VR/AR to practice more difficult operations, and then easily conduct the precise positioning of the operation site; in the military field, VR/AR can be used by the army to identify the operational environment and obtain the real-time geographical location data and other important data, as well as to military exercises and training [29], in the education field, VR/AR technology is used for training and practice as well as making VR/AR books; in the field of marketing, AR technology is used to show consumers the shape and characteristics of vehicle models via terminals in a virtuality-reality fusion way so as to promote vehicle sales.

Fig. 14.4 Applications of VR/AR engineering



14.3 Visualization for Modeling and Interactive Simulation

Hessam S. Sarjoughian

Visual artifacts provide potent means for the modeling, simulation, verification, validation, and accreditation life cycle stages. From hand-drawn to digital, multi-dimensional visualizations have served as part and parcel of capabilities for creating, maintaining, executing, and evaluating simulation models. Compared with descriptive and mathematical modeling, visual modeling is uniquely placed to help create useful models for a wider range of users. Visual abstractions can lead to reducing time and effort, for example, in defining models, developing and executing experiments, and evaluating observed structures and behaviors of simulation models.

Primitive and compound visual objects, which are generally more accessible to human cognition and thinking, are instrumental for existing and advancing concepts, theories, methodologies, frameworks, and practices. As a result, visualization has steadily become a critical part of models' development, especially for component-based modeling theories and their corresponding frameworks and tools [32]. This is evident as information at different levels of abstraction and distinct aspects best supports different modes of human problem understanding, problem-solving, and decision-making. Indeed, a myriad of static and dynamic graphical representations have been developed and targeted for modeling and simulation communities at large and numerous user communities as diverse as archeologists, biologists, climatologists, educationists, immunologists, neuroscientists, physiologists, and roboticists. Engineering and the arts have long been the beneficiaries of visualizations to simplify and elaborate concise complex problem formulations and complicated solutions.

Models can range from conceptual to semi-formal and formal representations. In principle, they can be formulated as descriptive, mathematical, and visual artifacts or actual proxies of real-world processes. Simulations enabled with visual objects

offer unique benefits for the development, operation, and use of actual and hypothetical complex systems-of-systems, including cyber-physical systems. Visualizations further the reach and impact of simulation for learning, training, and maintenance of real-world computational and physical systems to modeling and simulation practitioners.

Visualization can present operational purposes (e.g., supply and demand flow), causal structures (e.g., information dependency), system operations (e.g., electricity transmission), physical parts (e.g., electric grid components), and processes (e.g., feedback control) [33]. The use of visualization where concrete details are illuminated via abstract organizational, functional, and purposes for simulations is significant [34]. Insights via visual notations of the parts and their connections in whole models at varying resolutions can be gained. Graphical representations enabled with flexible hierarchical organizations reveal direct/indirect constraints for observing intricate simulation models that may have linear, superdense, or combined time data sets benefiting from visual behavioral debugging to visual analytics [35].

Early simulation tools, such as Matrixx [36] supporting hierarchical block diagram modeling with linear time plots and STELLA [37] supporting graphical system dynamics modeling, showed the advantages and better development of simulation models using different and complementary visual notations and user interfaces. Such capabilities continue to become even more critical due to the importance of formal composable heterogeneous modeling [38]. Very large-scale models naturally lead to needing simulation models with bigger data sets, heterogeneous hierarchical structures, and multimodal behaviors. Numerous frameworks and tools have been developed by employing different types of structurally organized shapes and lines with corresponding control and view graphics for user manipulation and viewing. Visualizations to support the creation, execution, and evaluation of heterogeneous models remain at the forefront of research challenges facing modeling and simulation science, engineering, and practice [39].

14.3.1 Visual Component Modeling

Visual modeling serves the needs of science, engineering, and humanity and increasingly, the topics that blend them for highly complex systems of systems. Of especial importance has been to serve highly diverse, sophisticated, and evolving modeling communities and stakeholders. In recent years, visual modeling has been gaining a greater prominent role in all aspects of simulation-based scientific and engineering research, innovation, and practice. It has become invaluable as a uniquely powerful means for developing models, conducting simulations, verifying models, validating simulations, and using them in operational settings.

From a broad point of view, general-purpose textual and visual languages, frameworks, and tools are commonly used for model representations, displaying simulation results, and user interfaces for evaluations and decision-making. Data visualization has been serving two classical complementary roles. One is for the post-simulation output data analytics. Another is for observing and interacting with

executing simulations. The former is for the data to be represented as charts, geo-location maps, controls, object graphs, and forms in dashboards [40]. The latter is for facilitating simulation executions (e.g., using standalone and Web-services computing platforms) as well as for managing simulation experiments (e.g., behavior exploration) targeting different audiences.

A key capability beyond those noted above is to develop simulation models visually. Visual modeling refers to the graphic representation of systems of interest using general-purpose and domain-specific graphical modeling languages. Visual models promote human understanding and comprehension of complex ideas and thus contribute to higher quality models at a faster development time and lower cost, especially when used with model syntax and supported with well-defined execution (operational or denotational) semantics. Visual modeling can simplify and increase communication and collaboration among stakeholders having varying backgrounds and skills. This is evident in the widely in-use, open, and standardized general-purpose visual languages, including UML and SysML, that are commonly used across many modeling and simulation communities. Behavior modeling is commonly specified using state machines, sequence, and activity diagrams [41].

Visual modeling affords concepts, methods, frameworks, languages, and tools to enrich the creation of different kinds of simulation models in any existing or conceivable application domain. Visual modeling can support any kind of modeling such as agent, declarative, functional, spatial, and more broadly component-based continuous, discrete-time, and discrete-event model types. General-purpose languages, frameworks, and tools are used to present data, structural, and behavioral abstractions using different widget types with a mixture of texture, geometric, and sound data. These are built using frameworks and tools such as the business intelligence reporting tool [42] and data driven documents [43].

Domain-specific languages (DSL) specialized for numerous kinds of application domains exist using many programming languages supported with built-in graphical passive and active widgets. An example framework is developed for waste management systems [44] based on model-driven architecture (MDA) [45] and frameworks supporting graphical modeling [46]. The visualization studio modeling SDK has a DSL definition diagram for specifying hierarchical metamodels and the graphical notation for domain-specific languages. The resulting application framework has an underlying formal language enabling domain expert modelers to build, configure, and manipulate wastewater management systems flexibly. Such visual languages are well-suited for conceptual modeling, where entities in the real world have transparent and one-to-one visual counterparts. Because conceptual modeling languages map directly from entities in the real world to visual objects, they are easier and more natural to use for developing modeling and simulation frameworks and tools. Recent visual programming languages built using the Eclipse modeling framework (EMF) [47] with the graphical modeling framework offer capabilities that make the development of DSLs simpler.

The combination of visualization capabilities for modeling, simulation, and evaluation is instrumental in identifying and addressing the why, what, and how questions tied to the uncertainty and stochasticity common to dynamic, large,

complex hybrid systems [48, 49]. They help determine the purposes of every simulation by defining the model's scope, the details in the built simulation, and the simulation results to be observed and evaluated subject to many external stimuli and experimental conditions. Conceptualization, design, implementation, testing, and debugging of intricate structural and behavioral separately and together benefit from visual modeling.

14.3.2 Visual Interactive Simulation

Visual modeling naturally leads to visually rich interactions with executing simulations. The visual abstraction of modeled objects provides useful decision support together with the capability for automated model checking and coded generations. Visual interactive simulation (VIS) supports symbolic manipulation, interactions, and offline analysis [50]. Dynamic visual displays, changing the model during simulation runs, linking dynamic input/output time trajectories, and histograms creating graphic replicas of actual and future systems to be built. Such capabilities are aimed at user-friendly systems that mesh human judgment and computation to improve the decision-makers' effectiveness. Users can use benefit from visual analytics to gain insight into simulation data sensitivity and as well as identify and dynamically interact with any model components for special transient changes needed essential for validation of complex models requiring parallel and/or distributed execution [35].

Visualization of models and their simulations requires identifying data sets from big data, changing data resolution, and selecting displays with varying operational modes. They help users in calibrating models useful for any purpose for which simulations are to be used. It is easy to observe the value of general- and special-purpose dynamic, multi-layer visual interactive interfaces. They allow identifying problems with data, models, and experiments, understanding from very small to very large data continuum, forming hypotheses, and discovering emergent dynamics [51]. Effective simulation configurable displays should account for subtleties in data content, data heterogeneity, data coherency, and limited spaces for statistical, descriptive, and exploratory analyzes [52]. In general, it should be noted careful analysis is needed to choose and aggregate data well-suited for displays while allowing intuitive controls for multi-layer and multi-aspect user interfaces.

14.3.3 Visualization Designs

Effective visualizations should serve multiple purposes, although they can become demanding due to requiring extensive computational power and complex information and controls to enable manipulations of phenomena, comprehension, and evaluation. Visualization capabilities are being expanded to help explore the structural relationships and behavioral evolutions of phenomena using machine learning and other methods [53].

Concrete methods can be undertaken to develop powerful visualizations to complement the development of abstract, computable models [54]. First, descriptions of alternative ways for users (use-cases and use-case scenarios) to achieve their purposes are identified. Second, suitable abstraction levels for data sets of interest are defined. Third, controls for executing and viewing simulation trajectories are devised. Fourth, the static and dynamic data and control in view of software architectures such as Model-Façade-View-Control architecture are formulated. Fifth, use-case scenarios are used to minimize the total number of displays and controls, thus leading to simpler and more effective user interfaces and visual analytics. Visual modeling with interactive simulation can lend itself to formulating and understanding causal and logical scalability and complexity traits, especially to address ambiguity inherent in the modeling, simulation, design of experiments, and evaluating predicted outcomes in time and space.

References

1. SEBoK Editorial Board (2020) The guide to the Systems Engineering Body of Knowledge (SEBoK), v. 2.2. In: Cloutier RJ (eds) The Trustees of the Stevens Institute of Technology, Hoboken, NJ. Last accessed on 29 Sept 2020. www.sebokwiki.org
2. INCOSE (2017) Systems Engineering Vision 2020, version 2.03. International Council on Systems Engineering (INCOSE), INCOSE-TP-2004-004-02
3. Estefan JA (2008) Survey of model based systems engineering methodologies. International Council of Systems Engineering, San Diego, CA
4. Gianni D, D'Ambrogio A, Tolk A (2014) Modeling and simulation-based systems engineering handbook. CRC Press
5. Durak U, Ören T (2016) Towards an ontology of simulation systems engineering. In: Proceedings of the SCS spring simulation multi-conference, Pasadena, CA, USA
6. Loper ML (2015) Modeling and simulation in the systems engineering life cycle. Springer, Berlin
7. Rainey LB, Tolk A (2015) Modeling and simulation support for system of systems engineering applications. Wiley
8. Mittal S, Martín JLR (2013) Netcentric system of systems engineering with DEVS unified process. CRC Press
9. Yilmaz L, Ören T (2009) Agent-directed simulation and systems engineering. Wiley-VCH
10. Mittal S, Diallo S, Tolk A (2018) Emergent behavior in complex systems engineering: a modeling and simulation approach. Wiley
11. Risco Martin JL, Mittal S, Ören T (2020) Simulation for cyber-physical systems engineering: a cloud-based context. Springer, Berlin
12. IEEE (2010) IEEE recommended practice for distributed simulation engineering and execution process (DSEEP), IEEE Std 1730-2010
13. IEEE (2010) Standard for modeling and simulation high level architecture—framework and rules, IEEE Std 1516-2010
14. D'Ambrogio A, Durak U (2016) Setting systems and simulation life cycle processes side by side. In: Proceedings of the 2016 IEEE international symposium on systems engineering (ISSE 2016), October 3–5, Edinburgh, UK
15. Atkinson C, Kuhne T (2003) Model-driven development: a metamodeling foundation. *Software IEEE* 20(5):36–41
16. OMG (2003) MDA Guide, revision 2.0 (ormsc/14-06-01)

17. Bocciarelli P, D'Ambrogio A, Giglio A, Paglia E (2019) Model-driven distributed simulation engineering. In: Mustafee N, Bae K-HG, Lazarova-Molnar S, Rabe M, Szabo C, Haas P, Son Y-J (eds) Proceedings of the 2019 Winter Simulation Conference (WSC 2019), December 8–11, 2019, National Harbor, USA
18. D'Ambrogio A, Petriu D (2011) First international workshop on model-driven approaches for simulation engineering (Mod4Sim'11). www.sel.uniroma2.it/mod4sim11
19. Topcu O, Durak U, Oguztuzun H, Yilmaz L (2016) Distributed simulation: a model driven engineering approach. Springer
20. D'Ambrogio A, Bocciarelli P, Delfa J, Kisdı A (2020) Application of a model-driven approach to the development of distributed simulations: the ESA HRAF case. In: Proceedings of the 2020 Spring Simulation Conference (SpringSim 2020)
21. IEEE (2015) ISO/IEC/IEEE International Standard-Systems and software engineering—System life cycle processes. ISO/IEC/IEEE 15288:2015
22. INCOSE (2015) Systems engineering handbook: a guide for system life cycle processes and activities. In: International Council on Systems Engineering (INCOSE), INCOSE-TP-2003-002-04, Wiley
23. Zhao Q, Zhou B et al (2016) A brief survey on virtual reality technology. *Sci Technol Rev* 34 (14):71–75
24. Yin RM, Li BH, Chai XD (2007) Image-based rendering techniques in virtual reality: a survey. *J Syst Simul* 19(19):4353–4357
25. Azuma RT (1997) A survey of augmented reality. *Presence Teleoperators Virtual Environ* 6 (4):355–385
26. Dahne P, Karigiannis JN (2002, October) Archeoguide: system architecture of a mobile outdoor augmented reality system. In: IEEE Proceedings. International symposium on mixed and augmented reality, pp 263–264
27. Li BH, Chai XD, Zhang L, Li T, Qing DZ, Lin T, Liu Y (2018) Preliminary study of modeling and simulation technology oriented to neo-type artificial intelligent systems. *J Syst Simul* 30(2):349–362
28. Zhao Q (2011) 10 scientific problems in virtual reality. *Commun ACM* 54(2):116–117
29. Van Krevelen DWF, Poelman R (2010) A survey of augmented reality technologies, applications and limitations. *Int J Virtual Reality* 9(2):1–20
30. Berg LP, Vance JM (2017) Industry use of virtual reality in product design and manufacturing: a survey. *Virtual Reality* 21(1):1–17
31. Ong SK, Yuan ML, Nee AYC (2008) Augmented reality applications in manufacturing: a survey. *Int J Prod Res* 46(10):2707–2742
32. Rhyne T-M, Chen M (2013) Cutting-edge research in visualization. *Computer* 46(5):22–24
33. Rasmussen J, Pejtersen A, Goodstein L (1994) Cognitive systems engineering. Wiley
34. Bennett KB, Flach JM (2011) Display and interface design: subtle science, exact art. CRC Press
35. Sarjoughian HS, Sundaramoorthi S (2015) Superdense time trajectories for DEVS simulation models. *SpringSim (TMS-DEVS)*, pp 249–256
36. Richmond B (1985) STELLA: software for bringing system dynamics to the other 98%. In: International conference of the system dynamics society. Keystone, CO, USA, pp 706–718
37. Walker R, Gregory C, Shah S (1982) MATRIX x: a data analysis, system identification, control design and simulation package. *IEEE Control Syst Mag* 30–37
38. Sarjoughian HS (2006) Model composability. In: Proceeding of the 2006 winter simulation conference. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, pp 149–158
39. Li JK, Takanori F, Kesavan S, Ross C, Mubarak M, Carothers CD, Ross RB, Ma K-L (2019) A visual analytics framework for analyzing parallel and distributed computing applications. *IEEE visualization in data science*. Vancouver, Canada, , pp 1–9
40. Few S (2006) Information dashboard design: the effective visual communication of data. O'Reilly Media, Inc.

41. Sarjoughian HS (2017) Restraining complexity and scale traits for component-based simulation models. In: Winter simulation conference. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, pp. 675–689
42. BIRT (2018) Business intelligence reporting tool. <http://www.eclipse.org/birt/>. Retrieved from <http://www.eclipse.org/birt/>
43. Bostock M, Ogievetsky V, Heer J (2009) D³ data-driven documents. *IEEE Trans Visual Comput Graph* 17(12):2301–2309
44. Zarrin B, Baumeister H (2014) Design of a domain-specific language for material flow analysis using Microsoft DSL tools: an experience paper. In: Proceedings of the 14th workshop on domain-specific modeling. Portland, Oregon, United States, pp 23–28
45. Object Management Group (2014) MDA Guide Version 2.0. Retrieved from <https://www.omg.org/mda/>
46. Fuhrmann H, von Hanxleden R (2010) Taming graphical modeling. In: Petriu DC, Rouquette N, Haugen Ø (eds) International conference on model driven engineering languages and systems. Springer, Berlin, Heidelberg, pp 196–210
47. Steinberg D, Budinsky F, Merks E, Paternostro AM (2008). *EMF: eclipse modeling framework*, 2nd edn. Pearson Education
48. Zeigler BP, Sarjoughian HS (2017) *Modeling and simulation of systems of systems*, 2nd edn. Springer, London
49. Kellner M, Madachy R, Raffo D (1999) Software process simulation modeling: why? what? how? *J Syst Softw* 46(2–3):91–105
50. Bell PC, Taseen AA, Kirkpatrick PF (1990) Visual interactive simulation modeling in a decision support role. *Comput Oper Res* 17(5):447–456
51. Ware C (2019) *Information visualization: perception for design*. Morgan Kaufmann
52. Tufte ER (2001) *The visual display of quantitative information*. Graphics Press
53. Isaacs KE, Gimenez A, Jusufi I, Gamblin T, Bhatele A, Schulz M, Hamann B, Bremer P-T (2014) State of the art of performance visualization. In: Proceedings of Eurographics conference on visualization, pp 141–160
54. Rouse WB (2015) *Modeling and visualization of complex systems and enterprises: modeling and visualization of complex systems and enterprise*. Wiley



Supporting Social Science and Management Areas

15

Paul K. Davis 

Abstract

This chapter of the SCS M&S Body of Knowledge addresses two topics, namely, using causal modeling and simulation to enhance aspects of social science and using causal models to aid managers and other decisionmakers. To this end, it discussed simulation approaches supplementing traditional social science approaches, particularly agent-based generative models.

Keywords

Modeling and simulation · Social science · Causal variables · Decision support · Generative modeling · Inverse generative modeling

15.1 Background: Contrasts of Paradigm and Semantics

Paul Davis.

This article addresses two topics: (1) using causal modeling and simulation (M&S) to enhance aspects of social science and (2) using causal models to aid decision-makers. The first section sets the stage by contrasting how social scientists and physical scientists approach modeling and define terms. Section 15.2 discusses how modern causal M&S can supplement traditional social-science methods. Section 15.3 discusses how M&S can better aid strategic-level decision-making.

P. K. Davis (✉)

The RAND Corporation and the Pardee RAND Graduate School, Santa Monica, CA, USA
e-mail: pdavis@rand.org

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2023
T. Ören et al. (eds.), *Body of Knowledge for Modeling and Simulation*,
Simulation Foundations, Methods and Applications,
https://doi.org/10.1007/978-3-031-11085-6_15

373

Table 15.1 Contrasts

Term	Class A. Most physical science and engineering	Class B. Most social science and data science
Theory	Coherent characterization of a system or broad phenomenon based on both empirical knowledge and logic <i>Example: Newton's laws, Quantum Mechanics</i>	A relatively general expression of relationships within a domain with an empirical basis <i>Examples: Equilibrium economics or Prospect Theory in psychology</i>
Model	(a) the application of theory to a problem context; (b) a useful but simplified version of theory; (c) a synonym for theory	A hypothesized relationships in a given context; or notional relationships to be viewed skeptically; or a synonym for theory
Explanation	Description of how elements of a system affect each other, either at a time or in generating system behavior over time <i>Examples: statistical-dynamics explanation of thermodynamics; Newton's explanation of solar eclipse</i>	The fraction of variance predicted by the main terms of a regression <i>Examples: demographic explanation of election results; explanation of Supreme Court verdicts based on its political composition</i>
Causal variables	System variables that, if changed, can change other system variables <i>Example: Flattening an object will cause increased drag and a slower rate of fall when the object is dropped in the atmosphere</i>	Observable variables that, if changed, are associated with subsequent changes in other observable variables <i>Example: A teenager's zip code predicts his or her future wealth</i>
Prediction	System behavior expected from a causal model and its inputs <i>Examples: Expected braking of an automobile; the expected safe landing of the Apollo Lunar Lander in 1969</i>	System behavior expected based on a historical-statistical model <i>Example: Predicted frequency of fierce storms based on a regression model using historical data</i>

Stark differences of paradigm and terminology exist between two classes of research. Class A is traditional physical science and most branches of engineering. Class B is traditional social science, data science, and some aspects of engineering (e.g., machine learning). Table 15.1 draws some contrasts. The dichotomy is imperfect, exceptions exist, and some researchers straddle the classes, but understanding the distinctions is important. The following paragraphs elaborate on the items of the table.

15.1.1 Theory

To those trained in physics and classic engineering, theory is revered—it is what pulls together knowledge coherently in causal terms. Social science, despite some unifying themes [1], is more fragmented [2]. Social scientists are accustomed to having many competitive and even contradictory theories in a particular domain.

They are comfortable with this and attempt to choose the theory that is most useful to them. They are often skeptical about attempts to develop unifying theory because contextual matters are so crucial in social phenomena. In the current era of “big data,” some see empirical analysis as superior to theory and refer scornfully to theory, by which they have in mind what might be termed bad but pretentious theory, as when economic theory is derived rigorously from axioms that do not realistically describe the real world (Bookstaber [3], p.184).

15.1.2 Models

A model is best seen as an application of theory to a particular problem and context, or as a useful simplification of theory. In practice, the terms “theory” and “model” are often used synonymously. In the physical sciences, however, models describe cause-effect relationships and sequences. In social science, they usually refer to a statistical association.

15.1.3 Explanation

Physical scientists explain phenomena with cause-effect relationships and sequences.

Social scientists’ concept of “explanation” refers to how much of the scatter in data is accounted for by the explicit terms of their statistical model. Even Occam’s Razor is used differently across communities. To a social scientist, applying Occam’s Razor means preferring as few regression variables as needed to fit the data without over-fitting. Physical scientists have in mind Einstein’s interpretation of Occam’s Razor: using a model that is “as simple as possible, *but not simpler*” [emphasis added]. This means including variables that theory sees as important even if some of the variables are not easily measured and even if they have small effects for the particular cases on which empirical data exists.

15.1.4 Causal Variables

Although the social sciences are currently dominated by statistical methods, they would benefit from a rebalancing giving relatively more emphasis to causal models [2]. The argument for this has been made especially well by Judea Pearl, who notes the inability to discuss fundamental questions without resorting to causal models (e.g., questions such as “Why?” and “What if...?”) [4, 5]. Ideally, inquiry includes a mix of theory-driven and data-driven approaches [6].

15.1.5 Prediction

Physical scientists see a predictive model as giving an accurate forecast of something before it happens (e.g., before next year’s eclipse). A social scientist, however, may

Table 15.2 Dimensions of model validity

Dimensions	Interpretation
Description	Identify salient structure, variables, and processes
Causal explanation	Identify causal variables and processes and describe reasons for behavior in corresponding terms
Postdiction	Explain past behavior quantitatively with inferences from causal theory
Exploratory analysis	Identify and parameterize causal variables and processes; estimate approximate system behavior as a function of those parameters and variations of model structure (coarse prediction)
Prediction	Predict system behavior accurately and even precisely (as assessed with empirical information)

refer to a model as predictive if it describes accurately existing data that was not used in creating the model (the out-of-sample data). That is confusing because the model might very well not be predictive about future events if the system changes.

15.1.6 Evaluating Models

How models should be judged also varies across discipline and subdisciplines. It is counterproductive to argue about a model's validity without distinguishing among very different dimensions, as suggested in Table 15.2 [2]. A similar breakdown refers to the uses of models as reason, explain, design, communicate, act, predict, and explore (Page [7], p. 15).

As an aside, an overlapping but different set of dimensions draws distinctions based on specification level. Does a model and the system it purports to describe agree at the level of the observation frame, input/output behavior, input/output function, state transition, or coupled components? This framework has been applied particularly for discrete-event and continuous simulation [8].

15.2 Supplementing Traditional Methods of Social Science (and Other Type-A Endeavors)

It is one thing to use M&S as part of social-science inquiry. It is quite another to use it to aid strategic decision-making. Section 15.2.1 touches lightly on classic attempts to do so. Section 15.2.2 discusses newer themes, such as dealing with the special challenges of complex adaptive systems

15.2.1 Classic System Methods

M&S has been used since the 1950s for problem areas involving individual and social behaviors. Only a few aspects of this are mentioned here.

System Dynamics

“System dynamics” refers generically to how systems change over time. “System Dynamics” (with capitals) originated with Jay W. Forrester at the Massachusetts Institute of Technology. It has been refined and nurtured for decades [9, 10, 11] and has a noted textbook [12], an international organization (the System Dynamics Society), a journal (*System Dynamics Review*), and a Web site with extensive resources <https://www.systemdynamics.org>. It has been applied successfully in countless studies. Although System Dynamics suffered from controversy after a 1972 study on the Limits to Growth (see the 30-year update [13]), the study’s insights and even its predictions have held up well [14, 15].

By and large, System Dynamic models reflect social-behavioral considerations indirectly through deterministic influences on stocks and flows, feedbacks, and the parameter values thereof. With rare exceptions, it does not include agents.

Soft System Thinking

Another strand of system thinking has attempted to recognize human aspects of social-behavioral phenomena that fit uncomfortably in frameworks originating in rational-actor economics, control theory, or engineering. Historically, these have been more influential in Europe than the United States, as with the work of Peter Checkland [16] and texts on “soft OR” [17], but some American efforts are well known [18], sometimes under such different labels as methods for “learning organizations” [19]. As a related example of how trends may be changing, at least one graduate school of policy analysis is changing its core curriculum to reflect the nature of complex adaptive systems and lessons learned from past often-failed efforts to intervene effectively in such systems [20].

Multiple Types of Modeling

One consensus has long been that social scientists and policy analysts should employ a wide range of modeling methods, depending on the nature and context of their work [7, 21]

15.2.2 Newer Themes

15.2.2.1 Qualitative Modeling

Social scientists understand a great deal about phenomena that resist precise description because the variables in question are often hard to observe directly, because hidden variables create variation, and for other reasons. Nonetheless, much can be described well with *qualitative* models—as in system diagrams, tables, or other mechanisms [16, 22]. Narrative analysis can be seen as a powerful use of qualitative modeling [23]. So also, human gaming (as distinct from game theory) can be seen as a kind of M&S that lacks some kinds of rigor (e.g., precise reproducibility), but can illuminate such factors as player values, mental models used by the players, and the ways in which humans adapt creatively to circumstances [24, 25].

15.2.2.2 Agent-Based Generative Modeling (and Inverse Generative Modeling)

The advent of agent-based modeling (ABM) was a particularly consequential development for the use of models in social science. An early book by Joshua Epstein and Robert Axtell sketched a revolutionary vision for simulating societal phenomena [26]. The authors noted that we need models that truly *explain* social-behavioral phenomena and that this requires the modeler to be able to *generate* the observed behavior in detail [26, 27], preferably from the bottom up. Much progress has been made in ABM technology since then. It has become accessible to researchers with relatively simple languages such as NetLogo [28] and more heavy-duty systems such as RePast-Symphony [29]. Many platforms exist with different strengths and weaknesses [30]. A good Web site of ABM-related resources exists with particular emphasis on economics <http://www2.econ.iastate.edu/tesfatsi/>.

At this point, a plethora of generative models exist that purport to explain social-behavioral phenomena. A deep problem, however, is that different agent-level rule sets can generate the same macroscopic behavior. How do we better understand which of these rule sets is best? That is the challenge of “inverse generative modeling.” An important start on meeting that challenge is described in the Agent Zero work of Joshua Epstein [31].

15.3 Aiding Strategic Decision-Making

If M&S is to be effective in decision-aiding on problems with social-behavioral aspects, as distinct from merely being a rich source of insight for science, it needs to be designed accordingly. In particular, it needs to reflect the major advances that have occurred over the last 25 years in dealing effectively with uncertainty and disagreement. Table 15.3 summarizes related admonitions [32]. The following sections elaborate on three themes: dealing with uncertainty, designing models for exploratory analysis, and seeing simulation as a mechanism for helping policy makers learn and prepare, rather than as a tool of prediction.

15.3.1 Dealing with Uncertainty and Disagreement

A core concept of much modern work has been seeking courses of action likely to perform well across a wide range of assumptions, i.e., in any of a wide range of futures. This is by contrast with seeking to optimize a plan for a particular set of assumptions. It is also in contrast with considering uncertainties on the margin by varying assumptions one at a time. Exploratory analysis may, for example, test a strategy across the full space of cases or scenarios generated by changing all model inputs simultaneously. It was difficult to pursue such a strategy 30 years ago when it was first proposed [33], but the technological hurdles have tumbled. Equally important, the concepts and analytical techniques have improved.

Table 15.3 Admonitions about making M&S useful in aiding decision-making

Admonition for social-behavioral modelers and analysts	Comment
Begin with campaign plan for analysis and decision-aiding	Plan to exploit diverse methods, tools, and information sources. Include interactive methods for discovery, testing, and education
Use system approach to comprehend the whole and to frame the problems within it	See the whole and interactions of its parts. Include all relevant factors, variables, and processes
Use causal models	Understand, reason, and communicate
Recognize different classes of system state: use different models or variable structure models	Account for changes in character of system, including emergent phenomena
Use data to test, inform, and calibrate causal models	Use theory-informed approach to data analysis, but also encourage competition to challenge existing theories and allow data-driven approaches to show their virtues
Construct multi-resolution models or model families	View issues at deeper level to avoid blunders and at higher level to see simplifications
Design from outset for exploratory analysis of uncertainties and disagreements	Understand where intervention is and is not feasible, controllable, and valuable; understand vulnerabilities; anticipate possible problems; plan for adaptation (requires ability to explore at low resolution and to zoom as appropriate)
Generate uncertainty sensitive outputs	Go beyond “what-if? Questions” with region plots, tradeoff curves, phase diagrams, and other devices
See decision-aiding as helping people understand the system, game board, and moves, rather than to predict the future	Move from predict-and-act paradigm to paradigm of exploring possibilities and planning for flexibility, adaptiveness, and robustness (FARness) (planning for adaptiveness for short)

These have applied not just to narrow planning, but to strategic planning that involves choosing among alternative *portfolios* of action necessary for strategic planning [34, 35]. In recent years, the newer approaches have been pursued under the rubric of Robust Decision-making (RDM) [36] and have exploited advances in model composition, data mining, and other techniques. A recent book provides an overview and a sampling of past applications related primarily to climate change and water management [37]. A related international organization, the Society for Decision making Under Deep Uncertainty, is young and vibrant. Its Web site is <http://www.deepuncertainty.org>.

Two admonitions from Table 15.3 perhaps merit elaboration. The first is the need to design *from the outset* for broad multidimensional uncertainty analysis. Although it is easy enough with standard simulations to conduct traditional sensitivity analysis, it is quite another matter to conduct exploratory analysis across the multidimensional input space.

15.3.2 Designing for Exploratory Analysis

What does such designing-from-the-outset look like in M&S? It likely includes multi-resolution modeling (MRM) so that relatively simple models can be used for broad exploration, followed by more detailed exploration on selected matters as necessary [32]. Increasingly, it should also include ambitious architectures with self-aware and self-adapting characters [38, 39].

A core concept here is that policy makers need to be able to *reason* about the decisions they are making [35]. They will seldom want merely to follow the advice of some unfathomable algorithm. To reason, however, they need a model with a small number of top-level variables (e.g., 3 or 5 rather than 10, 100, or 1000). This puts a premium on relatively simple high-level causal models. The simplified models are good for broad, exploratory analysis, but not for understanding deeper issues. For that, one needs analysis that zooms into detail where detail is warranted. In some cases, decision aids can be designed for real-time zooming into detail. In other cases, the detail can be provided as back-up material. It should, however, be available at the time policy makers are reasoning, so that if they have penetrating questions, those questions can be answered immediately.

Having such MRM capability is often feasible in applications, but it is far easier to attain if the capability is sought from the outset and the models are built accordingly. Further, it is essential to recognize that the lower resolution models are necessarily *approximations*, which should be tailored to the application and its context. Being able to generate such good approximations requires deeper knowledge and an understanding of how to project deeper knowledge into a lower-dimensionality depiction. Human beings do such skillful aggregation constantly in everyday life without being consciously aware of doing so. It is relatively uncommon, however, for modelers to think much about it. Instead, they may assume, for example, that simple averaging is a good aggregation. Systematic methods for “good” context-dependent aggregation have yet to be developed and built into tools.

15.3.3 Designing Decision Aids with Education and Experience-Building in Mind

It is tempting to believe that M&S should aid policy makers by providing accurate predictions upon demand and, with that in mind, to build detailed data bases with the best available information and models representing the best estimate of how the world works.

When dealing with complex systems, however, and particularly complex adaptive systems (CAS), that approach is misguided. Uncertainties and disagreements dominate—about data and even about model structure [40]. Just as exploratory analysis is preferred to point prediction, so also decision-aiding should help the policy maker navigate, orient, and operate despite such complications [41, 42]. This is akin to having flight simulators when training pilots: No one knows

precisely what circumstances a pilot may encounter, so the pilot should be good at characterizing circumstances and adapting to those that arise. Such skills can be obtained with “many hours on the simulator.” Interestingly, strong policy makers are often comfortable with this: They understand that uncertainties make firm predictions implausible and that they will have to make recurring decisions with only partial knowledge and with a variety of constraints reflecting disagreements and conflicting considerations at the time. With experience, however, they can become much better at navigating such turbulent seas. Lest this point be missed, the reader should note how different this paradigm is from running a computer model for best-estimate cases and presenting results to the policy maker.

References








1. Nyblade B, O'Mahony A, Sieck K (2019) State of social and behavioral science theories. In: Davis PK, O'Mahony A, Pfautz J (eds) *Social-behavioral modeling for complex systems*. Wiley, Hoboken, NJ, pp 63–99
2. Davis PK, O'Mahony A (2019) Improving social-behavioral modeling. In: Davis PK, O'Mahony A, Pfautz J (eds) *Social-behavioral modeling for complex systems*. Wiley, Hoboken, NJ, pp 15–48
3. Bookstaber R (2017) *The end of theory: financial crises, the failure of economics, and the sweep of human interactions*. Princeton University Press
4. Pearl J (2009) *Causality: models, reasoning, and inference*. Cambridge University Press, Cambridge, Massachusetts
5. Pearl J, Mackenzie D (2018) *The book of why: the new science of cause and effect*. Basic Books, New York
6. Sliva A, Really SN, Blumstein D, Peirce G (2019) Combining data-driven and theory-driven models for causality analysis in sociocultural systems. In: Davis PK, O'Mahony A, Pfautz J (eds) *Social-behavioral modeling for complex systems*. Wiley, Hoboken, NJ, pp 311–336
7. Page SE (2018) *The model thinker: what you need to know to make data work for you*
8. Zeigler BP, Muzy A, Kofman E (2019) *Theory of modeling and simulation (Third edition): discrete event and iterative system computational foundations*. Academic Press
9. Forrester JW (1971) *World dynamics*. Wright-Allen Press, Cambridge, MA
10. Forrester JW (1963) *Industrial dynamics*. MIT Press, Cambridge, Massachusetts
11. Holstein JA, Gubrium JF (2011) *Varieties of narrative analysis*. SAGE
12. Sterman JD (2000) *Business dynamics: systems thinking and modeling for a complex world*. McGraw-Hill, Boston
13. Meadows DH, Randers J, Meadows DL (2004), *The limits to growth: the 30-year update*. Chelsea Green, White River Junction, Vermont
14. Bardi U (2011) *The limits to growth revisited*. Springer, New York
15. Turner G (2008) *A comparison of the limits to growth with thirty years of reality*. CSIRO, Canberra, Australia
16. Checkland P (1999) *Systems thinking, systems practice (includes a 30-year retrospective)*. Wiley, Chichester, England
17. Rosenhead J, Mingers J (eds) (2002) *Rational analysis for a problematic world revisited: problem structuring methods for complexity, uncertainty and conflict*, 2nd edn. Wiley, New York
18. Ackoff RL (2008) *Ackoff's best: his classic writings on management*. Wiley, New York
19. Senge PM (2006) *The fifth discipline: the art & practice of the learning organization*. Penguin Random House, New York

20. Davis PK, McDonald T, Pendleton-Julian A, O'Mahony A, Osoba O (2020) A complex systems agenda for influencing policy studies, WR-1326. RAND, Santa Monica, CA
21. Cioffi-Revilla C (2014) Introduction to computational social science: principles and applications. Springer-Verlag, London
22. Davis PK, O'Mahony A (2017) Representing qualitative social science in computational models to aid reasoning under uncertainty: national security examples. *J Defense Model Simul* 14(1):1–22
23. Forrester JW (1969) *Urban dynamics*. Wright Allen Press, Cambridge, Massachusetts
24. Caffrey MB (2017) *On wargaming*. Naval War College Press, Newport, Rhode Island
25. Perla P, Curry J (eds) (2012) *Peter Perla's the art of wargaming a guide for professionals and hobbyists*. lulu.com
26. Epstein JM, Axtell RL (1996) *Growing artificial societies: social science from the bottom up*. MIT Press, Cambridge, Massachusetts
27. Epstein JM (1999) Agent-based computational models and generative social science. *Complexity* 4(5):41–60
28. Wilensky U, Rand W (2015) *An introduction to agent-based modeling: modeling natural, social, and engineered complex systems with Netlogo*. MIT Press, Cambridge
29. North MJ, Macal CM (2007) *Managing business complexity: discovering strategic solutions with agent-based modeling and simulation*. Oxford University Press, USA
30. Abar S, Theodoropoulos GK, Lemarini P, O'Hare GMP (2017) Agent based modelling and simulation tools: a review of the state-of-art software. *Comput Sci Rev* 24:13–33
31. Epstein JM (2014) *Agent_zero: toward neurocognitive foundations for generative social science*. Princeton University Press, Princeton, New Jersey
32. Davis PK (2019) Lessons on decision aiding for social-behavioral modeling. In: Davis PK, O'Mahony A, Pfautz J (eds) *Social-behavioral modeling for complex systems*. Wiley, Hoboken, NJ
33. Davis PK (1994) Institutionalizing planning for adaptiveness. In: Davis PK, Monica S (eds) *New challenges in defense planning: rethinking how much is enough*. RAND Corporation, California, pp 73–100
34. Davis PK (2002) Analytic architecture for capabilities-based planning, mission-system analysis, and transformation, MR1513. RAND Corporation, Santa Monica, California
35. Davis PK (2014) Analysis to inform defense planning despite austerity. RAND Corporation, Santa Monica, California
36. Lempert RJ, Popper SW, Bankes SC (2003) *Shaping the next one hundred years: new methods for quantitative long-term policy analysis*. RAND Corporation, Santa Monica, California
37. Marchau VAWJ, Walker WE, Bloemen PJT, Popper SW (eds) (2019) *Decision making under deep uncertainty: from theory to practice*. Springer, Cham, Switzerland
38. Yilmaz L (2019) Toward self-aware models as cognitive adaptive instruments for social and behavioral modeling. In: Davis PK, O'Mahony A, Pfautz J (eds) *Social-behavioral modeling for complex systems*. Wiley, Hoboken, NJ, pp 569–586
39. Yilmaz L, Ören T (2009) *Agent-directed simulation and systems engineering*. Wiley-VCH
40. Davis PK, Popper SW (2019) Confronting model uncertainty in policy analysis for complex systems: what policymakers should demand. *J Policy Complex Syst* 5(2):181–201
41. Rouse WB (2015) *Modeling and visualization of complex systems and enterprises*. Wiley, Hoboken, NJ
42. Rouse WB (2019) Human-centered design of model-based decision support for policy and investment decisions. In: Davis PK, O'Mahony A, Pfautz J (eds) *Social-behavioral modeling for complex systems*. Wiley, Hoboken, NJ, US, pp 798–808



Philosophy and Modeling and Simulation

16

Andreas Tolk , Ernest H. Page, Valdemar Vicente Graciano Neto , Paul Weirich , Nico Formanek , Juan Manuel Durán , Jean François Santucci , and Saurabh Mittal 

Abstract

This chapter of the SCS M&S Body of Knowledge summarizes philosophical foundations that are only partially addressed in other chapters in a constrained manner. It starts with a philosophical discussion of simulation epistemology,

A. Tolk (✉)

The MITRE Corporation, Charlottesville, VA, USA

e-mail: atolk@mitre.org

E. H. Page

The MITRE Corporation, McLean, VA, USA

e-mail: epage@mitre.org

V. V. Graciano Neto

Federal University of Goiás, Goiania, Brazil

e-mail: valdemarneto@ufg.br

P. Weirich

University of Missouri, Columbia, MO, USA

e-mail: WeirichP@missouri.edu

N. Formanek

High Performance Computing Center, Stuttgart, Germany

e-mail: nico.formanek@hlsr.de

J. M. Durán

Delft University of Technology, Delft, Netherlands

e-mail: j.m.duran@tudelft.nl

J. F. Santucci

University of Corsica, Corte, France

e-mail: santucci@univ-corse.fr

S. Mittal

The MITRE Corporation, Dayton, OH, USA

e-mail: smittal@mitre.org

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2023

T. Ören et al. (eds.), *Body of Knowledge for Modeling and Simulation*,

Simulation Foundations, Methods and Applications,

https://doi.org/10.1007/978-3-031-11085-6_16

383

including some idea about the role of ontologies as well. A timeline on scientific research and method development shows how simulation contributes to scientific research methods. This leads to the section dealing with the challenge of what type of knowledge can be acquired from simulation—the core question of epistemology. Additional sections investigate criteria for acceptance and hypothesis/proposing explanation in simulations. After working out differences of simulation and experience as well as simulation and experiments, the chapter concludes with observations on M&S as a discipline.

Keywords

Modeling and simulation · Epistemology of simulation · Ontology for simulation · Scientific method · M&S as a discipline

16.1 Philosophical Discussion of Simulation Epistemology

Andreas Tolk and Ernest H. Page

The discussion of philosophical topics only recently resurfaced within the professional modeling and simulation (M&S) societies. The ideas themselves are not new at all and have deep roots, often interwoven with discussions about ethics and the responsibility of the simulationist to clearly communicate the opportunities, but also to constraints and dangers of simulation, and today, this is more important than before, as our simulations are becoming more and more realistic, and users without a strong computational background can often no longer distinguish between the real and the virtual world, experiments conducted in both, and what they mean for knowledge generated from conducting and evaluating this experiments. Epistemology is the branch of philosophy that copes with the question of how to gain new knowledge within a discipline. It is deeply connected with ontology, the question of how to capture knowledge. This first subsection on epistemology will introduce the concepts and definitions and dive a little bit deeper into the importance of ontology and epistemology for simulation and vice versa.

16.1.1 Concepts and Definitions

This chapter is interested in the use of ontology and epistemology in support of M&S. Both terms are derived from the realm of philosophy but have also strong computer science applications. Ontology and epistemology pose the questions of what we know and how we can gain knowledge. As such, more than just a short definition is in order.

16.1.1.1 Ontology

Ontology is the study of being or the study of what exists. According to Kienzle [1], the concern of ontology is with reality: What are things made of, how many kinds of things are there, and what is the relation between mind and matter? According to Feibleman [2]:

Ontology is the widest system in any finite set of systems. It would perforce have to be an abstract body of knowledge and make the claim to truth. This could be either a tentative or an absolute claim. Its own terms of description are the categories of traditional metaphysics. The definition of Bentham, that the field of ontology is the field of supremely abstract entities, refers to these categories; in modern logical and mathematical systems we would call these categories the undefined terms employed in the unproved propositions which constitute the postulates of the system. There is no official ontology and contending ontologies must support their claims on the basis of the same criteria used by other kinds of system: consistency, completeness, and applicability. Rival ontologies exist theoretically and practically and assert both abstractly and concretely their respective claims.

From this excerpt, it can be observed that ontology is the description of reality. As we are discussing philosophy in this section, the question what reality itself is may have to be addressed as well. Very generally, the concept of reality can be divided in positivistic and post-positivistic views. Positivism assumes that we experience reality and can understand it through experiments and collection of empirical data. In post-positivistic science views, reality is understood as unknowable and indescribable, exemplified by the case of the supreme abstract entities, and we have access to different realities which are described under different ontologies. The wider the ontology, the closer the ontology is to be to reality, and a claim made in this ontology is going to be considered an absolute claim. Although each ontology for each model must be consistent, we cannot assume that all model ontologies describing the same reference also are derived from one common ontology that describes one reality. They may differ substantially, in particular in soft science (but, as we know, also in micro- and macro-scale of physical sciences).

Ontologically, we perceive in substantive or process terms [3]. Substantive ontology focuses on describing what something is, in terms of its parts and relations among parts. Process ontology focuses on describing how something is done. However, we describe processes in substantive terms. This can be explained from the perspective that the only way of observing how something is done is by observing the states of how something is transformed. Yet, this transformation is still a description of the thing itself and not on the undergone process.

16.1.1.2 Epistemology

Epistemology is the study of how we come to know. Epistemological beliefs are individual's belief about the definition of knowledge, how knowledge is constructed, how knowledge is evaluated, where knowledge resides, and how knowledge occurs. Epistemology seeks to answer the question: What can be considered knowledge, and how do we derive or create new knowledge? The answer to this question, formulated by the ancient Greeks, is still in debate today with a no clear consensus in sight. Epistemologically, a satisfactory definition of knowledge is that

of justified true belief “with conditions.” These conditions remark where justified true belief cannot be considered knowledge. Pragmatically, epistemology focuses on the validity of knowledge, considering, its sources, how it is justified, and under what conditions claims can truly be considered knowledge.

Epistemologically, we come to know through empirical or rational means. Empirically, we come to know through correspondence; what an individual perceives through his/her senses and it can be proven scientifically or accepted through pragmatical means is accepted as knowledge. Rationally, we come to know through coherence; what an individual creates in his/her mind, whether initiated by observation, and can be explained within a system of premises is accepted as knowledge. Both currents are accepted in the body of knowledge, and both have supporters and detractors. Biologists and experimental physicists, for instance, abide by seeking knowledge through empirical means. Mathematicians and M&S researchers, on the other hand, abide by seeking knowledge through rational means. For the conceptualization of a model, the epistemological constraints can therefore easily become very relevant when it comes to identification and selection of compassable solutions.

16.1.2 Ontology and Simulation

The archives of professional simulation societies comprise many contributions on ontology and simulation. However, most of them are more technologically than philosophically oriented. They apply ontologies to describe simulation systems or capture application domains using ontological means that help the simulation engineer to get access easier to this domain. As discussed in Robinson et al. [4], this use of ontologies is a valuable contribution, as it allows to unambiguously capture the result of the conceptualization process leading to a simulation. After all, these ontologies address methodological as well as referential questions [5]. Methodological ontologies capture knowledge about “how to model,” which means they focus on modeling paradigms, techniques, and formalisms. Conversely, referential ontologies capture knowledge about “what to model,” which means they model the referents and relations of the application domain.

There are a couple of contributions that shall be used as examples of current research and development efforts important for the body of knowledge. This enumeration can neither be exclusive nor complete and requires to be updated from time to time.

- Benjamin et al. [6] are among the first to publish about the possibility to address technical challenges in distributed simulation modeling using a generalizable ontology-based framework instead of a domain-specific solution. This work was inspired by the earlier discussion on composability of simulation components, which had been proven to require more transparency of the components to be composed.

- Examples for methodological ontologies are the Discrete-event M&S Ontology (DeMO) [7] or the Component-oriented Simulation and Modeling Ontology (COSMO) [8].
- The System Entity Structure (SES) provides a high-level ontology introduced for knowledge representation of decomposition, taxonomy, and coupling of systems based on system-theoretic principles [9]. The proximity to the simulation formalism DEVS, captured in its own sections of this compendium, allows the structural and dynamic representation of systems using SES describing the structure, and DEVS the dynamic of the components as well as the composed system. SES is applied in several recent research efforts in many application domains, such as cyber-physical systems.

Simulations are based on models. These models are a purposeful abstractions and simplifications of a perception of reality; it is the result of the conceptualization. Ontologies are conceptualizations as well, so they are strong candidates to play a stronger role within the M&S domains. Gruber [10] defined an ontology to be an “explicit specification of a conceptualization,” and Borst [11] defined an ontology even stronger as a “formal specification of a shared conceptualization.” The value of formal specifications is that they support the construction of proofs regarding underlying properties of a specification, so that in many cases validity of formal specifications can be mathematically proved, plus they are machine readable. (Per the incompleteness theorem of Gödel, there can be true properties of a specification that cannot be proven. In that sense, validity is not provable per se. However, the general insolvability of this problem should not be used as an excuse not to do it where possible.) As such, they can provide for the transparency required for the conceptual alignment of conceptual representation, as it has shown to be needed for new concepts, such as M&S as a Service.

16.1.3 Epistemology of Simulation

The question on how to create knowledge using M&S has been discussed for at least two decades, but predominantly in the philosophy of science community. The use of models is well established in the scientific community. Scientists have always used models of the world to capture knowledge. Over time, in properly organized processes of inquiry, they modify the models, making them better, adding attributes or relations, implementing new processes, and overall improving accuracy and fidelity. In his philosophical primer, Gelfert [12] observes that the heterogeneity of models in science and the diversity of their uses and functions are nowadays widely acknowledged.

The use of simulation, however, is not as widely recognized as a possible epistemological method or tool. Winsberg [13] was among the first scientist to systematically deal with this challenge. A workshop on epistemology of simulation (EPOS) was organized by Professor Frank and Professor Troitzsch at the University of Koblenz in 2004. This workshop was among the first attempts to bring natural

and social scientists, philosophers, and computer scientists together to address their common epistemological and methodological issues [14]. Several EPOS workshops followed, and their results were evaluated by Grune-Yanoff and Weirich [15] in a systematic review. At the same time, Winsberg [16] summarized his research on these topics in his book. Within the Society for Modeling and Simulation, an expert panel during the Winter Simulation Conference [17] introduced some of these ideas to the broader community.

The core ideas of epistemology of simulation can be summarized as follows. Computer simulations are computer programs. As such, they follow the constraints and limitations of computation, including fundamental insights as captured in the Church-Turing thesis or Gödel's incompleteness theorem. Simulations cannot overcome them. However, simulation allows us to capture concepts by its axioms and rules in an executable form. This forces scientist and researchers to be very precise when formulating theories. When doing so, computational science is the result. Ken Wilson's discoveries about phase changes in materials using simulation earned him a Nobel Prize in physics [18]. The discovery of the Higgs boson elementary particle [19], in which simulation was successfully used to guide experiments to allow for successful observations, is another example of computational science success. The simulation comprised all aspects of the guiding theory and therefore could help to look exactly where the theory predicted certain events to occur so that the scientists could focus their observations of the real world accordingly.

These core ideas also hint to three potential pitfalls when using simulation-based experiments in lieu of real-world experiments. First, simulations capture all the available knowledge in an executable form, but that implies that only what has been captured will also be executed. The simulation-based experiment cannot reveal some new insight into a phenomenon that was not captured by the algorithms or the input data for the simulation. Therefore, if an important factor, relation, or process required for a discovery is not modeled, the simulation cannot support the desired discovery, which leads to the second pitfall, hermeneutical mistakes. Scientists using simulation-based experiments must constrain their interpretations to what is in the model and not what they expect following their worldview. While epistemological mistakes happen when we do not model important aspects of the research questions, hermeneutical mistakes interpret something into simulation results that is not included in the model. Finally, simulation-based experiment can result in regress, which is a series of statements in which a logical procedure is continually reapplied to its own result without approaching a useful conclusion. When we test a hypothesis by implementing it as a simulation and then use the simulated data in lieu of empirical data as supporting evidence justifying the propositions, we are actually only proving that we build a correct simulation of the hypothesis, not that the hypothesis itself is true.

It is the task of the simulation engineer to ensure that these pitfalls are avoided and the simulation is only applied and interpreted within its validity constraints. This requires a solid understanding of the philosophical underpinnings of simulation that could only be touched in this short summary.

16.2 Scientific Research Methods and Simulation

Andreas Tolk

The introduction to the epistemology of simulation in the previous section provided the philosophical foundation on how to gain knowledge using simulation. In this section, a short historical timeline on scientific research methods through the lens of simulationists is given before evaluating further how simulation is used in support of scientific research today.

16.2.1 A Timeline of Scientific Research Method Developments

Scientific research methods adapted and improved over the centuries. Our understanding of the scientific method is relatively young, as described by Goldman [20]. It generally is understood to have started 500 years ago, when two scientists shaped our understanding of what science knows, and how we can increase this knowledge: Sir Francis Bacon in England and René Descartes in France.

Sir Francis Bacon (1561–1626) is considered the father of the inductive-empirical method. He postulated that knowledge can only be derived by observation and data collection. Only that which can be observed should be the subject of science. Parallel to these efforts, René Descartes (1596–1650) defined the deductive-rational method. It assumes that a world order of physical laws can be described by mathematical models that are the basis of knowledge. Observations and data validate knowledge.

These two competing visions of the scientific methods, also known as Baconian Empiricism and Cartesian Rationalism, were unified in the work of Sir Isaac Newton (1642–1727). His work on “Mathematical Principles of Natural Philosophy” (original 1687 in Latin, Newton et al. [21]) was the standard literature for more than 200 years and is still the foundation of high school physics. Newton utilized mathematical models that allowed for coherent and consistent interpretation of observations. In other words, he used models for the conceptualization of experience generalizing observations and validated them by empirical observations. The work of Newton was continued by John Locke (1632–1704) and David Hume (1711–1776), who introduced the ideas of skepticism and challenges of inductive reasoning. They asked the question: On what evidence can we assume that our experiences are universal or even repeatable in the future? Immanuel Kant (1724–1804) reintroduced a kind of certainty and stability with his cosmological theory. His “Universal Natural History and Theory of the Heavens” (original 1755, Kant and Jaki [22]) provided a renewed foundation for rationalism.

The nineteenth century focused on the application of science within many engineering efforts. Theory often took a backseat to application. At the same time, scientist started to focus on basic principles. In the twentieth century, Bertrand Russell (1872–1970), Henri Poincaré (1854–1912), and Percy Bridgman (1882–

1961) agreed on the principles that scientific knowledge is not about reality, but focuses more on common concepts, how to measure them, and how to express them. Mathematical logic was the common language to express these concepts. Science was interpreted as a commonly conceptualized and experiential world, expressed by models based on logic. Their work became the foundation of logical positivism, embracing the idea that only science can provide a rational account of the world. In their view, truth was driven by a physical-deterministic account of reality.

This viewpoint was shaken by Albert Einstein (1879–1955) and Niels Bohr (1885–1962) with their relativity theory and quantum theory. The work of Werner Heisenberg (1901–1976) and Erwin Rudolf Josef Alexander Schrödinger (1887–1961) added to the new challenges: Scientific theories had become so complex and often argued against empirical observations of the daily life. As a result, scientific theories were no longer understood as a converging process getting closer and closer to reality, but a discontinuous and even revolutionary process, in which old theories were replaced by new ones in a non-convergent but progressive process [23].

The philosophical underpinnings of the philosophy of science were also shaken during this period. After the devastating events of two world wars, the waning influence and often perceived loss of credibility of religious worldviews, and even disappointment with alternative secular worldviews, several philosophers of science started to question the objective role of science to search for reality and serve mankind by allowing scientific progress. Michel Foucault (1926–1984), Jacques Derrida (1930–2004), and Jean- François Lyotard (1924–1998) provided the justification for postmodernism in their work, which arose in various forms but can be characterized by radical skepticism of the possibility of obtaining objective knowledge. Relativism and constructivism, the idea that scientific knowledge is a mere social construction and knowledge must always be understood in a social context, influenced not only the social sciences, but quickly took root in the humanities and the broader philosophical foundations. The ideas did lead to the development of critical theories, that mainly address the misuse of science and the scientific method in favor of currently ruling classes and the need to deconstruct “scientific findings” that are rooted in too much bias resulting from such misuse [24].

However, despite the criticism of variants of the scientific method, the role of modeling has not diminished. The definition of modeling as a task-driven purposeful simplification and abstraction of a perception of reality holds under all paradigms and allows for communication between scientists across the borders of paradigms and belief systems. As stated by Robert [25]: “I have been, and remain, entirely committed to the idea that modeling is the essence of science and the habitat of all epistemology.”

16.2.2 The Increasing Role of Simulation with the Scientific Methods

While modeling as the process to provide a conceptualization was essential to the scientific methods since Newton, computer simulation had to wait for the necessary computational power to be available, which started to build up since the middle of the last century. As simulation allows to execute the conceptualization, this also allowed to bring theories to life. As discussed in Tolk [26], computational sciences are applied in archeology, biology, chemistry, materials science, economics, electromagnetics, engineering, finance, fluid dynamics, forensics, geophysics, history, informatics, intelligence, law, linguistics, mathematics, mechanics, neuroscience, particle physics, physics, sociology, statistics, and more. They all have in common that they use a conceptualization of their field to execute them; in other words, they are conducting simulations. The discovery of the Higgs boson particle was already mentioned in the last section. A more recent example is the use of supercomputers to analyze Covid-19 data leading to significant supporting observations for the Bradykinin hypothesis [27]. Using the Summit supercomputer at Oak Ridge National Lab in Tennessee, scientists analyzed more than 40,000 genes from 17,000 genetic samples trying to better understand the effects of Covid-19. Models of how the virus enters the human body and affects and utilizes human enzymes were used to compute effect predictions that could be verified by observation. The simulation shows how the SARS-CoV-2 virus consequently leads to the so far hypothesized Bradykinin Storm, which is an increased level of the bradykinin molecule in the cells, resulting in changes of diverse body functions on the organ level. These insights can explain many of the Covid-19 symptoms and even more important support the research for better treatment and even prevention.

In summary, while modeling has been tightly interwoven with the scientific method for centuries, the increase of computational power enabled the use of simulation to drive computational sciences and increased their role for gaining new insights as well as educating students using the power of immersive visualization of the underlying dynamics.

16.2.3 Modern Role of Models in the Scientific Method

The modern role of models in science has been described in detail by the philosopher of science Gelfert [12], who observes in his philosophical primer on models in the summary the following:

Whereas the heterogeneity of models in science and the diversity of their uses and functions are nowadays widely acknowledged, what has perhaps been overlooked is that not only do models come in various forms and shapes and may be used for all sorts of purposes, but they also give unity to this diversity by mediating not just between theory and data, but also between the different kinds of relations into which we enter with the world. Models, then, are not simply neutral tools that we use at will to represent aspects of the world; they both

constrain and enable our knowledge and experience of the world around us: models are mediators, contributors, and enablers of scientific knowledge, all at the same time. (Gelfert [12], p. 127).

In other words, models are used in the scientific community as mediators, contributors, and enablers of scientific knowledge, contributing to the comprehensive and concise representation of concepts, terms, and activities that make up the scientific domain, also known as the body of knowledge. They allow to comprehend, share, and reproduce research results, presenting theories in a more generally comprehensible and testable form. Modeling is intrinsically tied to scientific work.

It is this view that also shows the potential to support multidisciplinary team. Current big challenges often require that experts from multiple disciplines have to collaborate on a solution, and some of the disciplines differ significantly in the methods and tools, sometimes even terms used. The aspect of hybrid modeling, as discussed in other sections of this book as well, supports the use of different modeling paradigms in a coherent way to support a common research goal. Tolk et al. [28] showed how this approach can enable transdisciplinary research.

Finally, we are increasingly facing challenges that are defined by conditions “where analysts do not know, or the parties to a decision cannot agree on, (1) the appropriate conceptual models that describe the relationships among the key driving forces that will shape the long-term future, (2) the probability distributions used to represent uncertainty about key variables and parameters in the mathematical representations of these conceptual models, and/or (3) how to value the desirability of alternative outcomes [29].” Lempert and colleagues call this deep uncertainty, and the usual parametric variations and sensitivity analyses are not sufficient to address them. Marchau and colleagues [30] edited a recent book that provides various examples how to use families and hierarchies of models and simulations to conduct exploratory modeling and analysis to gain a better insight into the topology of the solution space under various assumptions.

16.2.4 Simulation-Based Experiments Supporting Scientific Methods

In one of the first sections of this book, Sect. 1.4.2, the entities of the Modeling and Simulation Framework were introduced. Although many textbooks start with the development of a conceptual model that is then transformed into a computer simulation, the experimental frame that encapsulates the source system, real or imagined, is pivotal when it comes to simulation-based experiments. A model is a task-driven abstraction and simplification, and it is the experimental that specifies the conditions under which the system is observed or experimented with. Like understanding the environment of a real-world experiment and excluding all unwanted effects from its context, so defines the experimental frame the context for the model. This is extremely important for simulation-based experiments, as the

model becomes the reality of the simulation: Whatever is excluded from the model cannot be observed or influence the simulation. If we exclude something important in our model, it cannot be observed in the simulation. This is an epistemological error. The counterpart is that we also cannot interpret simulation results by using something that has not been part of the context. This would be a hermeneutical error. Simulation experts must ensure that both kinds of mistakes are avoided: that everything important to address a challenge is modeled and that nothing is used in the interpretation of results that is not part of the model.

Despite these nontrivial challenges, simulation has matured to a point where it is widely accepted as an analysis and design tool complementary to theoretical considerations and experimental investigations, becoming the third pillar of gaining scientific knowledge. In his work, Ihrig [31] proposes a framework bringing theory building, simulation, and experimentation into the common context of epistemological insight, as captured in Fig. 15.1.

- In the theoretic setting, shown on the left side of the figure, propositions are derived from an existing theory, which also drives the design.
- In the experimental setting, shown on the right side of the figure, the real-world issue provides access to the empirical data and also is the source for the desired insights.
- In the simulation setting, the model provides the simulated data within the context of the simulation environment. Figure 16.1

Ihrig [31] shows that these settings provide a consistent reference frame to understand simulation experiments in the context of the conventional research approach, pointing also to the duality of simulation experiments: Looking at simulation from the theoretic settings, it is perceived as a possibility to conduct experiments. From the experimentation side, a simulation can be perceived as a theory capturing tool, opening the possibility to understand simulation systems as a good way to capture knowledge.

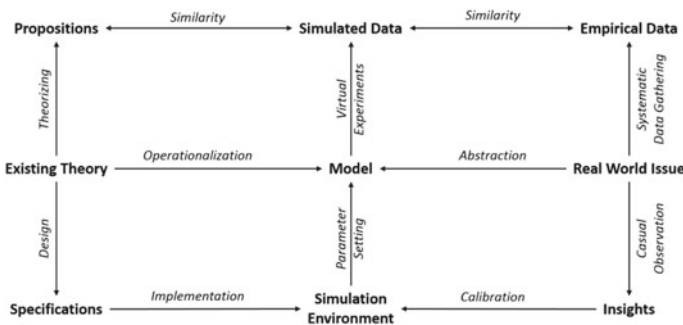


Fig. 16.1 Research Architecture proposed by Ihrig [31]

16.2.5 Conclusion on the Role of Modeling and Simulation

We started this section on “Scientific Research Methods and Simulation” with a short and simulation-focused view on the history of scientific methods, showing that models play an essential role in science general. We closed it with the observation that simulation can be justified to be part of the knowledge repository gained by such work.

However, a word of caution is in order as well. As discussed, each model is a simplification and abstraction of reality, and the model becomes the reality of the simulation. A simulation is therefore highly unlikely to represent all aspects of a systems, which is easily forgotten when the simulation drives a high-resolution, immersive environment that makes it feel very realistic. The context of validity is more important than ever, particularly when the user of the simulation is taking the results as face value.

Furthermore, there is the possibility of regress, as discussed in Tolk [32]. The danger of the simulationist’s regress is that predictions are based on a hypothesis, and then, the implementation of the hypothesis in form of the simulation system is used to conduct a simulation experiment that is then used as supporting evidence that the hypothesis is true. In other words, we test a hypothesis by implementing it as a simulation and then use the simulated data in lieu of empirical data as supporting evidence justifying the propositions: We create a series of statements—the hypothesis, the simulation, and the resulting simulated data—in which a logical procedure is continually reapplied to its own result! By assuming we are right, we show that we are correct! Particularly when addressing challenging problems in the social and philosophical realm, like social justice or the effects of climate change, or recently the effects of a pandemic and possible countermeasures, moral and epistemological considerations are deeply intertwined. In such conditions, it is human nature to cherry-pick the results and data that support the own worldview [33]. Simulationists are not immune to this, and they can implement their beliefs into a complex simulation system. This simulation system can then be used by others to gain quasi-empirical numerical insight into the behavior of the described complex system. As a result, the implemented worldview of the simulationist can easily be confused with a surrogate for real-world experiments.

In summary, the connections of modeling, simulation, and the scientific methods are stronger than ever before, and simulation is playing an increasing role in understanding and communicating results in this context. It is the role of the simulation expert to ensure that this is conducted within the context of validity, as captured in the code of ethics.

16.3 What Type of Knowledge Can Be Acquired from Simulation?

Valdemar Vicente Graciano Neto

Simulations come in many flavors and formalisms. The definition adopted in this BoK in Sect. 1.1 (scope) states simulations have three aspects: (1) perform experiments, (2) gain experience for training to gain/enhance skills, or for entertainment, and (3) imitation, pretense. Moreover, for scope purposes, this BoK delimits that solely experiment and experimentation aspects are within the scope of this study. A further extension on that concept says that “simulation is providing experience under controlled conditions for training, i.e., for gaining/enhancing competence in one of the three types of skills: (1) motor skills (virtual simulation or use of simulators), (2) decision and/or communication skills (constructive simulation such as business games, war games, or peace games; aka serious games), and (3) operational skills (live simulation). We then can sum this discussion of three aspects: knowledge, experience, and experiments. Experiments will be further elaborated in Sect. 16.4. Hence, this discussion will focus on knowledge and experience.

From a pragmatic point of view, *knowledge* involves (i) facts, information, and skills acquired through experience or education and the theoretical or practical understanding of a subject or (ii) awareness or familiarity gained by experience of a fact or situation [34]. However, from an information science perspective, knowledge is a more elaborated network of linked findings obtained from (scientific) discoveries and the highest form an information can assume [35].

Simulations have actually been source of knowledge from several sciences precisely by enabling the simulation consumer to (i) acquire skills, such as driving and piloting skills in airplanes or car simulators, (ii) reveal (theoretical or practical) understanding based on the elaboration of a simulation model or by data being consumed by the simulation (such as in black hole simulation), (iii) draw conclusions on a physical model, such as the feasibility of a structure in civil architecting and engineering simulators, (iv) predict the feasibility of organic structures, such as in simulation of synthesis of chemical molecules, (v) anticipate properties of a non-existent system, such as quality attributes of a novel spacecraft system, and (vi) predict patterns and emergent properties, such as birds grouping phenomenon.

The knowledge that can be acquired from a simulation is then related to the level of details and trustworthiness of the model being simulated. As detailed is a model as diversified is the number and types of information (and knowledge) that can be acquired from that simulation model.

16.4 Criteria for Provisional Acceptance in Science: Objectivity, Completeness, and Reproducibility

Paul Weirich

Acceptance in science is generally provisional because new information may prompt re-evaluation [36]. The criteria for acceptance depend on the type of candidate; the criteria for acceptance of a hypothesis differ from the criteria for acceptance of a model.

Acceptance has various senses [37]. In frequentist statistics, acceptance of a hypothesis occurs if the result of a statistical test does not lead to rejection of the hypothesis. Rejection occurs if the result's type is sufficiently improbable given the hypothesis. In another sense, acceptance of a hypothesis is voluntary endorsement of the hypothesis and entails belief that the hypothesis is true and a readiness to act on the basis of the hypothesis. The standards for acceptance of the hypothesis may depend on the gravity of the mistake of accepting the hypothesis when it is false and the mistake of not accepting the hypothesis when it is true [38].

In science, acceptance of a hypothesis is warranted only if the hypothesis is objective in the sense of being scientifically testable, and testing has yielded evidence that makes the hypothesis sufficiently probable considering the context, and so confirms the hypothesis [39]. Presentation of the hypothesis and its support should be complete, and acceptance is warranted only if the evidence supporting the hypothesis is extensive enough that collection of additional evidence is unlikely to undermine support for the hypothesis. Moreover, acceptance is warranted only if the evidence, if gathered from a test, is robust in the sense that the type of result obtained from the test recurs in repetitions of the test, and so is reproducible.

The criteria for acceptance of a model of a phenomenon depend on whether the model is proposed as an approximate description of the phenomenon, a means of approximately predicting the phenomenon, or a partial explanation of the phenomenon. For acceptance as an approximate description of the phenomenon, the criteria are similar to those for acceptance of a hypothesis. However, because a model may be useful despite some inaccuracy, a model may be accepted even if the evidence supports only its approximate accuracy. A predictive model may be accepted because it yields good predictions, even if the mechanism by which it obtains predictions does not correspond to the mechanism behind the phenomenon it treats. Its acceptance is warranted only given the model's validation by a track record of good predictions in cases not used to construct the model [40]. Acceptance of a model as a partial explanation of a phenomenon requires evidence that the model represents some features of the mechanism producing the phenomenon.

16.5 Hypothesis/Proposing Explanation in Simulation

Nico Formanek, and Juan Manuel Durán

Computer simulations are valuable instruments to explore a wide variety of scientific questions. But are they just instruments to compute analytically intractable models or do they change the way of scientific inference?

Philosophy of science has traditionally considered such questions under the label of *scientific explanation* (henceforth “explanation”), stating that the aim of science is to explain rather than just give descriptions. The goal was then to elaborate what it means that “X explains Y” (also called the explanatory relation between X and Y). Generally, it is hoped that at a suitably abstract level a description of explanations and reasons why some of them are better than others can be given. This should in no way be confused with the idea that such a description constitutes a norm for explanations and therefore scientific conduct. While in some cases it might help single out severely defective explanations, it is generally rather the other way round: The concept of explanation is abstracted from scientific practice.

Before we introduce some of the ideas what explanation could be, it is worth noting that one also needs to be concerned with the unit carrying out the explaining X (i.e., the *explanans*) and the unit to be explained Y (i.e., the *explanandum*). On a very general level, they are just propositions, statements, or sentences. On a more fine-grained level, theories, models, causal processes, and natural laws have variously been proposed as constituting the explanans, while phenomena of the real world are generally considered to be in want of explaining. As we will see later, this might change with the introduction of computer simulations into scientific reasoning.

Theories of explanation can roughly be divided into the following classes:

- Deductive/Nomological
- Causal/Mechanical
- Unificationist
- Pragmatic.

The deductive/nomological account of explanation is the oldest of the modern ones, dating back to 1940s [41], and the causal/mechanical one became popular during the 1970s [42], while the unificationist account had its heyday in the 1980s [43]. Every account has a pragmatic component. However, the label “pragmatic” here is reserved for the ones that count the explanation as a social or psychological process (e.g., Achinstein [44]). It what follows, we very briefly describe the four main accounts of explanation.

- *Deductive/Nomological Account:*

A scientific explanation is a deductive argument from a general law and certain other premises to the desired conclusion. The general law needs to be an essential premise. Deductively invalid arguments fail to explain, as do arguments not containing a general law.

- *Causal/Mechanical Account:*
A scientific explanation gives a description of the causal processes that lead to the production of the desired explanandum.
- *Unificationist Account:*
A scientific explanation maximally unifies the current set of scientific beliefs. Maximally unifying means to use as few and stringent argument patterns as possible to generate that set.
- *Pragmatic Account(s):*
These generally view scientific explanation as a social or psychological process. Conditions for successful explanation are sought in terms of those processes and thus highly context dependent. Pragmatic accounts are also sometimes called subjective to distinguish them from the other accounts which are considered objective.

Unsurprisingly many hybrids have been proposed that try to mitigate one or another problem of the pure accounts (for a review, see Salmon [45]).

Concluding this section, it should be noted that the general way to construct an account of scientific explanation proceeds through examples. This means that examples of good explanations from special sciences are examined and their shared properties are then made explicit. The deductive/nomological account for example is mainly derived from explanations found in physics, which was at that time considered a paradigm for science. So unsurprisingly, its treatment of explanation works well for physics and other highly mathematized sciences. Other accounts also trace their roots back to the special sciences from whose examples they were originally derived (e.g., the pragmatic account has a connection to social sciences). It can be argued that this mode of construction is too narrow to yield a general theory of explanation and some accounts tried to remedy that by being more abstract (e.g., the unificationist account). Nonetheless, their failings are mostly discussed by presenting counterexamples of purported explanations from ACMO special sciences which they are apparently unfit to deal with. The main aim is still to give an account of scientific explanation and not to give an account of explanation for a special science.

16.5.1 What Can Be Expected of the Explanatory Relation?

Giving an account of scientific explanation might be considered one of these philosophical exercises that does not have much impact on how science is done. In a sense this is very true, scientists in the past have echoed varied opinions if science explains at all. Duhem and Mach are probably the most famous naysayers, while the general consensus nowadays seems to be that science is the foremost purveyor of explanations. But the deeper issue at stake here is one of rationality. We would like to know which explanations are to be preferred above others or even to single out the scientific ones without referring just to contextual pragmatic facts. This is what is to be expected of a philosophical treatment of the explanatory relation. To

isolate the objective from the subjective conditions, to separate the pragmatic from the logical parts of explanation. What cannot be expected is some kind of automated method which could be algorithmically employed to generate or check candidate explanations.

If such an account can be given, it in turn can be used to justify and understand the use of computer simulations in science.

16.5.2 Description, Prediction, Confirmation, and Explanation

Apart from explanation, description and prediction have been offered as the main goals of science. Scientific theories should also be well confirmed, so all these things are epistemic goals of science. Philosophy of science has expounded their relative merits on various occasions. Explanation and description have often been contrasted to show that explanation offers knowledge gains beyond description. It is for example thought that having an explanation facilitates further theorizing about a phenomenon. Computer simulations can play a role in all of those processes, depending on the aim of the simulation. Like theories or scientific models, they can figure in prediction, confirmation, and explanation simultaneously. Take for example the stated aims of the Illustris project, a cluster of large-scale hydrodynamical simulations of the matter distribution and evolution in the universe. The “[...] ultimate goal in each case is a deepening of our understanding of the processes by which the observed population of galaxies formed and evolved over cosmic time.” [46], while reproducing and confirming existing theories are only thought of as ancillary.

So, while describing and predicting phenomena have their place in science, explaining or understanding them seems to have a special epistemic role. Unsurprisingly, the relation between understanding and explanation has been the subject of recent studies in philosophy of science (for a review, see Baumberger et al. [47]).

16.5.3 Can Computer Simulations Explain?

Most accounts of scientific explanation date back to times where computer simulation in science either did not exist or was not as pervasive as it is today. This led philosophers to question if and how they could incorporate it in the existing frameworks. The leading question is which role, if any, does computer simulation play in scientific explanation and the explanatory relation? It is to be expected that the answers depend on the account of scientific explanation. So far, computer simulations have been discussed using either the causal/mechanical or the unificationist account.

One strand of thought, which might be called tool view or “the received view of computer simulations” (see Durán [48]), has emerged that views simulations as purely instrumental to explanation. While they might help explore the consequences

of analytically intractable theories or models, they do not represent the target system in any sensible way because they introduce numerical approximations and errors. The causal mechanisms or natural laws at play can therefore not be part of a simulation, which thus is in turn unable to explain directly [49]. A similar argument was already leveled against theories or models containing idealizations (so basically every theory or model) most famously by Cartwright [50]. One strategy to cope with this objection is to claim that incomplete models or theories still partially explain or at least help identify and control for explanatory factors (see Mäki [51]). In the debate for computer simulation, this strategy culminated in the tool view by Krohs [49] and Weirich [52]. Computer simulations are viewed as tools to identify and control for explanatory factors in intractable models. They are not a necessary part of the explanatory relation and can, so to say, be discarded after their work is done. The (partial) explanations are still achieved only by models and theories.

Durán leveled an indispensability argument against this classification of computer simulations as mere tools and proposed that the unificationist account more accurately reflects the explanatory work simulations do [53]. Noting that the main use of computer simulations is in cases of extremely intractable models, he argued that they are our sole provider of knowledge about explanatory factors. So, we cannot identify which factors are in fact explanatory without the simulation. Thus, the simulation has to be an essential part of the explanatory relation, namely it explains its results. The *explanandum* is not a statement about the world but a statement generated by the simulation about the computational model and the world. In the same vein, the *explanans* is assumed to be the computational model and not theories, natural laws, or mathematical models.

Of course, the main epistemic goal is to gain some understanding of natural phenomena not the results of a computer simulation, and this is achieved by embedding the explanatory work of the simulation within a somewhat modified unificationist framework. It is claimed that successful explanations are achieved by maximally unifying the set of beliefs about theories, models, and simulations.

If this account can be defended for less intractable models remains to be seen.

16.5.4 Generating/Testing Hypotheses with Simulations

The received view in philosophy of science is that science is a human activity and so scientists come up with hypotheses and experimentally check them. While the thought of a fully automated science has been entertained before, it is rarely expounded in detail because of the alleged remoteness of the situation (see, e.g., Humphreys [54]). Hybrid situations where part of the scientific process is automated (perhaps by computer simulations) have indeed been duly considered.

Recently, the role of models as exploratory tools has been noted with some of the examples relying heavily on computer simulations (see Refs. [55, 56]). So, one might argue that if simulations are essential to explore the consequences of models, they in turn are exploratory tools. Unfortunately, their specific contribution to generate hypotheses in these cases has not been considered in the literature.

Simulations have often been compared to experiments in their role of testing hypotheses that are otherwise intractable (for a review, see Winsberg [57]), with some going so far to argue that computer simulations are experiments [58]—the caveat being here that “being an experiment” is seldom meant ontologically but often epistemically. So this would mean that computer simulations could in some sense at least be used to replace experiments. Of course, this violates the empiricist leanings of many philosophers and scientists (see Beisbart [59], who argue for the indispensability of experiments. It is out of questions that computer simulations are heavily used in statistical inference, and there have been questions especially about the epistemic role of Monte Carlo simulations. Beisbart and Norton, for example, hold that they are part of the statistical inference and cannot be considered experiments.

The somewhat construed situation where one can choose between an equally epistemically valuable simulation and experiment rarely or never happens in scientific practice. So it is more plausible to think of computer simulations as extending the scientific process rather than replacing parts of it.

A perhaps more interesting question with regard to the advent of machine learning methods is if they could be classed as computer simulations and what exactly they then would simulate. At least on the face of it, they seem to automate large chunks of the scientific process including hypothesis generation and testing, so perhaps what they simulate is part of the scientific process (see Jantzen [60]). As most of these methods are still in their early stages and widespread success has not yet been achieved, it is perhaps not surprising that little consideration is given to them in the recent literature on the epistemology of computer simulation. A recurring argument in the case of machine learning methods is that they eliminate the need for human intervention in hypothesis generation and testing. Even in the case of Unsupervised Learning, this is at least somewhat doubtful (see Hennig [61]), because the methods still need fine tuning by humans.

In conclusion, it can be said that we are certainly in a hybrid situation where computer methods like simulation and machine learning extend our epistemic grasp of the world, but a fully automated science seems still out of reach.

16.6 Experiments Versus Simulation

Valdemar Vicente Graciano Neto

Simulation has been faced as a platform for the conduction of experiments. 24 of the 100 definitions for the term *simulation* collected in Ören’s work associate simulations with experiments [62]. Experiments are a test done in order to learn something or to discover if something works or is true [63]. Also, scientists can evaluate the validity of an evaluated hypothesis by simulation.

Simulations are particularly valuable to support experiments in (i) a real rather than a virtual trial could be excessively expensive or dangerous, such as experiments on resistance of space probes or chemical reactions, such as in atomic bombs [64], (ii) the subject of study is inaccessible due to distance (galaxies) or dimensions (atoms or molecules), (iii) the system to be developed is not tangible, such as software, (iv) large scale and complex, such as a smart city, (v) optimization scenarios, where many combinations should be evaluated according to a pre-defined set of variables to be maximized or minimized, and (vi) social sciences in which ethics can prevent experiments, among other domains.

Results provided by simulation-based experiments are reliable if the morphism relation exists between the lumped and the base model and between the base model and the simuland. Moreover, the results should also be complemented with tests with the real final product to assure the required quality and precision in operation.

Studies have reported guidelines for experiments in software engineering research [65] and social sciences [66], and benefits from the use of simulation have been reported in several domains. Once experiments imply on the elaboration of hypothesis and observation in a controlled environment, the nature of the subject of investigation should be considered to accordingly select the variables and simulation formalisms that are suitable to that domain.

16.7 Experience Versus Simulation

Jean François Santucci

From the seventeenth century, models were more and more often inspired by laboratory experiments. Finally, it sometimes happens that they are deduced from theories themselves conceived without experience, thanks to a purely intellectual approach, the “thought experience” (the theory of relativity conceived by Albert Einstein in 1915 could only be experienced ‘in 1919). Today, several physical theories provide models that it has not yet been possible to validate by experience. For some, moreover, no experimental protocol will ever be able to fully test them. In addition, there is often a question of “digital experience” to underline the analogy between the practice of a simulation and the conduct of a physics experiment.

Experience and simulation can be defined using a spectrum of activities. At one end of the spectrum, those classic experiences are used when studying a given phenomenon. The hope is to learn something from the experience. On the other end of the spectrum, we can find simulations, which mimic a real-world scenario exactly.

Experiential learning sits in the middle of this spectrum and is involved in Artificial Intelligence.

Planning and decision problems belong to an area of Artificial Intelligence (AI) that aims to produce plans for an autonomous agent, that is, a set of actions that it must perform in order to reach a given goal from a known initial state. Because of

the stochastic nature of the problems, the planning and decisions very often lean on machine learning (ML) [67, 68]. However, the computer implementation of models and algorithms associated with ML may require lot of work because (i) it is difficult to select which model (algorithm) will fit with the problem to solve (there is a large choice of models allowing to develop a ML algorithm), (ii) once the model has been chosen, it is difficult to select the hyper-parameters of the chosen model that will allow to give good results after the learning phase, and (iii) classical tools dealing with ML do not allow to connect learning agents involved in ML with a powerful simulation environment. This lack of connection implies difficulties to deal with temporal aspects which may appear in the problem definition or to deal with multi-agent or even harder to deal with dynamical aspects as agents that may appear or disappear during simulation. For instance, the AI can help the “simulationist” in the modeling of complex systems that are impossible to represent mathematically [69–71]. On the other hand, M&S can help AI models failed to deal with complex systems for lack of simple or unworkable heuristics [72, 73].

AI learning techniques have already been used in a DEVS simulation context. Indeed, in [70] the authors propose the integration of some predictive algorithms of automatic learning in the DEVS simulator in order to considerably reduce the execution times of the simulation for many applications without compromising their accuracy. In [69], the comparative and concurrent DEVS simulation is used to test all the possible configurations of the hyper-parameters (momentum, learning rate, etc.) of a neural network. In [74], the authors present the formal concepts underlying DEVS Markov models. Markov concepts [75] of states and state transitions are fully compatible with the DEVS characterization of discrete-event systems. In [72], temporal aspects have been considered into Generalized Semi-MDPs with observable time and a new simulation-based RL method has been proposed.

Machine learning is a type of AI that use three types of algorithms (Supervised Learning, Unsupervised Learning, Reinforcement Learning) in order to build models that can get input set to predict an output set using statistical analysis. For the simulation part, Monte Carlo simulation [76] is often used to solve ML problems by using a “trial and error” approach. Monte Carlo simulation can also be used to generate random outcomes from a model estimated by some ML technique.

Simulation can also be useful in the context of Reinforcement Learning (RL) [77] and Markov Decision Processes (MDPs) [75]. For example, simulation can improve the experimental replay generation in RL problems where the agent’s experiences are stored during the learning phase.

RL components can be developed to replace rule-based models. This is possible when considering human behavior and decision-making. These learning components can either be used in simulation models to reflect the actual system or be used to train ML components. By generating the data sets needed to learn neural networks, simulation models can be a powerful tool in deploying the algorithms of recursive learning.

16.8 Simulation in the Spotlight of Cutting-Edge Technologies

Valdemar Vicente Graciano Neto

Simulations allow the anticipation and study of characteristics of systems under development, especially critical and large systems. Thus, simulations are particularly important for cutting-edge technologies, particularly smart cities, and other classes of system-of-systems, i.e., systems that are themselves formed as a set of other independent systems [78, 79].

Cutting-edge technologies such as novel space aircrafts, deployment of satellites, or smart cities can be too complex and expensive to be prototyped or deployed under trial conditions [80]. Hence, simulations can support a cheaper but enough precise analysis of these new systems to be built. Moreover, in most of the design problems and particularly regarding to cutting-edge technologies, the real system does not exist, yet, which motivates the use of simulations [64].

Several cutting-edge technologies are benefited by simulations. Cyber-physical systems (CPS), smart cities, deep learning and advanced Artificial Intelligence mechanisms, robotics, blockchain, superfast broadband, wireless power, nanotechnology, quantic computing, and brain-machine interaction are instances of such technologies that can be dangerous or expensive or impactful to be deployed without a reasonable prototyping. Simulations provide the means to predict their properties, enable visualization at large of microworlds, and safe prototypes.

16.9 Modeling and Simulation (M&S) as a Discipline

Saurabh Mittal

“Discipline” as defined by Oxford English Dictionary [81] is “a branch of learning or scholarly instruction.” Building on a list by Krishnan [82] for defining an academic discipline, we present the following factors and how M&S addresses it.

1. *A specific object of research:* There must exist a unique object or a relationship with other established objects that becomes a subject of study. There are certainly two objects here: modeling and simulation. These two objects have specific definitions. Both objects are associated with separate activities and generate unique artifacts.
2. *A body of accumulated specialist knowledge:* This factor identifies the specific knowledge set that is accumulated over a period, and it would take a considerable effort to arrive at that level of specialist skills. There are over 834 M&S books recorded in the United States Library of Congress [83], with 119 published in the last century and 714 published in the past two decades.

3. *Theory and concepts for organizing the specialist knowledge*: This factor documents the theory and various concepts that define the specialist knowledge. It puts together a unique paradigm and worldview. The theory of M&S originated in mid-1970s [84]. As simulation got pervasive and got applied to other domains, mechanisms for performing modeling and simulation activities and management of various artifacts were developed based on theoretical foundations leading to best practices.
4. *Nomenclature and technical language*: This factor defines a glossary, lexicon, and taxonomy that contributes to a vocabulary, a *lingua franca* for the specialist knowledge. Information about this aspect of M&S is available in Appendix A1 – Dictionaries of Modeling and Simulation (M&S), Cybernetics, and Systems. (Compiled by Tuncer Ören)
5. *Specific research methods and tools*: This factor pushes through the innovative disruption and documents various research methods and tools that are applied by the specialists to continue to advance the object under study and in turn create more tools that provide efficiency and increase productivity. As evident from the published literature in the past two decades, the application of M&S to various domains and disciplines has developed specific research methods and various domain-specific and domain-independent tools.
6. *Institutional manifestation in the form of subject taught at universities or colleges*: This factor establishes post-secondary, undergraduate, and graduate educational programs that disseminate the knowledge to the next generation for their inclusion in the specialist group. There are many universities today that offer graduate courses, as well as full degree programs on M&S. A comprehensive list of universities and research centers is available in Oren et al. [85].
7. *A group of professionals practicing the specialized knowledge*: This factor creates an environment where the specialists can organize and discuss the subject at hand on a regular basis. Through the establishment of societies, research groups, conferences, and symposia, specialists gather and share the cutting-edge developments and refine the established methods and knowledgebase. There exist many peer-reviewed journals such as Transactions of Society of Modeling and Simulation (SCS), Transactions of Association of Computing Machines (ACM), Simulation Modeling in Practice and Theory (SIMPAT) by Elsevier, and many others. Conferences like Summer Computer Simulation Conference (SCSC) have been in existence for over 60 years. Today, there are many other conferences all year round. For example, in the United States there are Spring Simulation Conference, Autumn Simulation Conference, and the Winter Simulation Conference. There are others that are located in Europe (European M&S Symposium) and in Asia as well.
8. *Pursuit of synergy with other established disciplines*: This factor explores the relationship of the discipline with other established disciplines. It identifies the boundaries of the new discipline and further explores how it could advance other established disciplines. This leads to transdisciplinary and multidisciplinary efforts that strengthen all the participating disciplines.

9. *Sustainment of the discipline at economic and political levels*: This factor integrates the discipline into the societal fabric at economic and political levels where it starts having impact at national and global levels and may even become established as an enabling technology defining the future of mankind.

The first simulation game may have been created as early as 1947 by Thomas T. Goldsmith Jr. and Estle Ray Mann. M&S has now been in existence for over 70 years [86]. It addresses all the nine factors as we have seen above. The current MSBoK that this article is contributing to brings together the M&S knowledgebase. The contributions by leading experts in MSBoK and the content herein cover the first seven factors. The eighth factor was recently documented in a book by Mittal, Durak, and Ören [87].

Synergies play a critical role in the evolution of disciplines. Contribution of simulation to a discipline “x” is called “simulation-based x.” Already, simulation-based science, simulation-based engineering, and many other simulation-based disciplines are important examples of contributions of simulation to other disciplines, making simulation a powerful infrastructure for them. Table 16.1 (adopted

Table 16.1 Simulation-based disciplines (examples).

Areas	Disciplines
Engineering	Simulation-based (all types of) Engineering (Chaps. 3, 4, 7, 8) Simulation-based Cyber-physical Systems (Chap. 5) Simulation-based Complex Adaptive Systems (Chap. 6) Simulation-based Physical Internet (Chap. 13)
Natural Science	Simulation-based (all types of) Science (Chap. 9) Simulation-based Cosmology Simulation-based Astronomy
Health Science	Simulation-based Health Care (Chap. 10) Simulation-based Pharmacology
Social Science and Management	Simulation-based Social Science (Chap. 11) (Behavioral science, psychology, demography, sociology, public administration, political science, archeology, environmental studies, etc.) Simulation-based Economics Simulation-based Enterprise Management (Chap. 12) Simulation-based Planning and Scheduling Simulation-based Optimization Simulation-based Policy Improvement
Information Science	Informatics Artificial Intelligence (Machine Intelligence) (Chap. 6) Software Agents Communication Library Science
Education/ Training	Simulation-based Education (Chap. 13) Simulation-based Training (Chaps. 10, 14) (including Health Care and Military Training)
Entertainment	Simulation-based Games

Adopted from Ören et al. [64]

from Ören et al. [64]) highlights an extended list of disciplines that benefit tremendously from simulation-based approaches.

A comprehensive review of these disciplines along with the impact M&S has on them is documented in [87]. The following enumeration gives a short overview of the chapters of the book [87]. Chapter 2 lays out the M&S national and global landscape. Chapter 3 introduces simulation in the classical engineering process. Chapter 4 presents the role of simulation in systems engineering. Cyber-physical systems (CPS), Internet of Things (IoT), and, further, complex adaptive systems as emerging system architectures come along with a new set of challenges. Simulation is being pronounced as a key tool in tackling the upcoming challenges of engineering such complex sociotechnical adaptive systems. Accordingly, Chapter 5 presents simulation-based CPS and IoT, and Chapter 6 explores simulation-based complex adaptive systems. In Chapter 7, readers can find a discussion about the role of simulation in software engineering, and Chapter 8 explores simulation in architecture. Natural and health sciences are also benefiting from the contribution of simulation in great extent. Chapter 9 presents simulation-based science, and Chapter 10 introduces simulation in health care and health education and training. In the field of social sciences and management, there are various applications of simulation. Chapter 11 elaborates on simulation-based social science, and Chapter 12 elaborates on simulation-based enterprise management. In the direction of learning, education, and training, while Chapter 13 presents simulation-based learning, Chapter 14 presents simulation in military training.

In addition to the disciplines listed in Table 16.1, some other disciplines are already using simulation-based approaches. They include experimental archeology and simulation-based cosmology.

For the ninth factor, Glotzer et al. [88] stated:

Today we are at a ‘tipping point’ in computer simulation for engineering and science. Computer simulation is more pervasive today – and having more impact – than at any other time in human history. No field of science or engineering exists that has not been advanced by, and in some cases transformed by, computer simulation. Simulation has today reached a level of predictive capability that it now firmly complements the traditional pillars of theory and experimentation/observation. Many critical technologies are on the horizon that cannot be understood, developed, or utilized without simulation.

As Page [89] reported, there is an estimated annual global market exceeding \$18B USD, despite difficulties in accurate quantification due to M&S ubiquity and its strong incorporation in various disciplines. The countries with the highest investment levels in training simulation and simulators include United States, Russia, China, India, and the United Kingdom. The heaviest investors in Product Life-Cycle Management (PLM) software are France, United States, and Germany. Asia-Pacific and Latin America are expected to have the highest growth in medical simulation, driven by India, China, South Korea, Singapore, Brazil, and Mexico. An important subset of total spending are investments relating to research and development (R&D). Within the government sector, M&S R&D investments are typically embedded within the enterprise Science and Technology (S&T) budget, or as part of the Research, Development, Testing, and Experimentation (RDT&E)

budget for new/developing systems. Again, direct measures are elusive, but within the United States alone, annual R&D spending easily exceeds \$100B USD [90]. With respect to investments by industry, in-depth market surveys for global industrial R&D funding are available. A 2016 assessment suggests that total global R&D investments approach \$2 T USD [91].

The U.S. Government, recognizing the contribution of M&S technology to the security and prosperity of the United States, declared M&S as a National Critical Technology in 2007 [92].

To summarize, M&S qualifies as a discipline with a strong academic, economic, research, technological, and political impact. Additional arguments and contributions are given by Tolk and Ören [93].

References

1. Kienzle HJ (1970) Epistemology and sociology. *Br J Sociol* 21(4):413–424
2. Feibleman J (1953) *Ontology*. Johns Hopkins Press, Baltimore, MD, p 1953
3. Rescher N (2000) *Process philosophy: a survey of basic issues*. University of Pittsburgh Press, Pittsburgh, PA, p 2000
4. Robinson S, Arbez G, Birta LG, Tolk A, Wagner G (2015) Conceptual modeling: definition, purpose and benefits. In: *Proceedings of the 2015 winter simulation conference (WSC '15)*. IEEE Press, Piscataway, NJ, USA, pp 2812–2826
5. Hofmann M, Palii J, Mihelcic G (2011) Epistemic and normative aspects of ontologies in modelling and simulation. *J Simul* 5(3):135–146
6. Benjamin PC, Patki M, Mayer RJ (2006) Using ontologies for simulation modeling. In: *Proceedings of the 2006 winter simulation conference*, pp 1151–1159
7. Silver GA, Miller JA, Hybinette M, Baramidze G, York WS (2011) DeMO: an ontology for discrete-event modeling and simulation. *SIMULATION* 87:747–773
8. Teo YM, Szabo C (2008) CODES: An integrated approach to composable modeling and simulation. In: *Proceedings of the 41st annual simulation symposium*. IEEE CS Press, pp 103–110
9. Kim TG, Lee C, Christensen ER, Zeigler BP (1990) System entity structuring and model base management. *IEEE Trans Syst Man Cybern* 20(5):1013–1025
10. Gruber TR (1993) A translation approach to portable ontology specifications. *Knowl Acquis* 5(2):199–220
11. Borst WN (1997) *Construction of engineering ontologies*. Ph.D. thesis, University of Twente, Enschede
12. Gelfert A (2016) *How to do science with models: a philosophical primer*. Springer International Publishing AG, Cham, Switzerland
13. Winsberg E (1999) Sanctioning models: the epistemology of simulation. *Sci Context* 12(2):275–292
14. Frank U, Troitzsch KG (2005) Epistemological perspectives on simulation. *J Artif Soc Soc Simul* 8(4)
15. Grune-Yanoff T, Weirich P (2010) The philosophy and epistemology of simulation: a review. *Simul Gaming* 41(1):20–50
16. Winsberg EB (2010) *Science in the age of computer simulation*. University of Chicago Press, Chicago
17. Tolk A, Ihrig M, Page EH, Szabo C, Heath BL, Padilla JJ, Suarez ED, Weirich P, Yilmaz L (2013) Epistemology of modeling and simulation. *Proceedings of the winter simulations conference*. IEEE, Piscataway, NJ, pp 1152–1166

18. Wilson KG (1989) Grand challenges to computational science. *Future Gener Comput Syst* 5 (2–3):171–189
19. Atlas Collaboration (2012) Observation of a new particle in the search for the standard model Higgs boson with the ATLAS detector at the LHC. *Phys Lett B* 716(1):1–29
20. Goldman SL (2006) *Science wars: what scientists know and how they know it*. Lehigh University, The Teaching Company, Chantilly, VA
21. Newton I, Motte A, Cajori F, Thompson SP (1955) *Mathematical principles of natural philosophy*, vol 34. Encyclopaedia Britannica
22. Kant I, Jaki SL (1981) *Universal natural history and theory of the heavens*. Scottish Academic Press, Edinburgh
23. Kuhn TS (1962) *The structure of scientific revolutions*. Chicago, IL
24. Thompson MJ (2017) Introduction: what is critical theory? In: *The Palgrave handbook of critical theory. Political philosophy and public purpose*. Palgrave Macmillan, New York NY
25. Rosen R (1998) *Essays on life itself*. Columbia University Press, New York NY
26. Tolk A (2018) Simulation and modeling as the essence of computational science. In: *Proceedings of the 50th summer computer simulation conference*. Society for Computer Simulation International, San Diego, CA
27. Garvin MR, Alvarez C, Miller JI, Prates ET, Walker AM, Amos BK, Mast AE, Justice A, Aronow B, Jacobson D (2020) A mechanistic model and therapeutic interventions for COVID-19 involving a RAS-mediated bradykinin storm. *Elife* 9:e59177
28. Tolk A, Harper A, Mustafee N (2021) Hybrid models as transdisciplinary research enablers. *Eur J Oper Res* 291(3):1075–1090
29. Lempert RJ, Popper SW, Bankes SC (2003) *Shaping the next one hundred years: new methods for quantitative, long-term policy analysis*, RAND Technical Report MR-1626. Santa Monica, CA, USA
30. Marchau VAWJ, Walker WE, Bloemen PJT, Popper SW (2019) *Decision making under deep uncertainty: from theory to practice*. Springer, Cham, Switzerland
31. Ihrig M (2016) A new research architecture for the simulation era. *Seminal contributions to modelling and simulation*. Springer, Cham, pp 47–55
32. Tolk A (2017) Bias ex silico: observations on simulationist's regress. In: *Proceedings of the annual simulation symposium, spring simulation multi-conference*. Article 15. Society for Computer Simulation International, San Diego, CA
33. Shermer M (2017) How to convince someone when facts fail: why worldview threats undermine evidence. *Sci Am* 316(1):69
34. Oxford (2019) *Knowledge: English dictionary, thesaurus, & grammar help*. Available at: <https://www.lexico.com/en/definition/knowledge>. Last access Aug 2019
35. Kebede G (2010) Knowledge management: an information science perspective. *Int J Inf Manage* 30(5):416–424
36. Bock P (2001) Chapter 5: An epistemological journey. In: *An getting it right: R&D methods for science and engineering*. Academic Press, San Diego
37. Weirich P (2004) Belief and acceptance. In: Niiniluoto I, Sintonen M, Wolenski J (eds) *Handbook of epistemology*. Kluwer, Dordrecht, pp 499–520
38. Hacking I (2001) *Probability and inductive logic*. Chapter 18, “Significance and Power”. Cambridge University Press, Cambridge
39. Titelbaum M (Forthcoming) *Fundamentals of Bayesian epistemology*. Chapter 6, “Confirmation”. Oxford University Press, New York
40. Deaton ML (2006) Validation of simulation models. In: Johnson N, Kotz S (eds) *Encyclopedia of statistical sciences*, 2nd ed. Wiley, Hoboken, NJ
41. Hempel CG, Oppenheim P (1948) Studies in the logic of explanation. *Philos Sci* 15(2):135–175. <https://doi.org/10.1086/286983>
42. Salmon WC (1984) *Scientific explanation and the causal structure of the world*. Princeton University Press. <https://doi.org/10.2307/j.ctv173f2gh>

43. Kitcher P (1989) Explanatory unification and the causal structure of the world. <http://conservancy.umn.edu/handle/11299/185687>
44. Achinstein P (1983) *The nature of explanation*. Oxford University Press
45. Salmon WC (2006) *Four decades of scientific explanation*. University of Pittsburgh Press
46. Illustris—About. Retrieved 14 Mar 2021, from <https://www.illustris-project.org/about/#public-three>
47. Baumberger C, Beisbart C, Brun G (2017) What is understanding? An overview of recent debates in epistemology and philosophy of science. In: *Explaining understanding. new perspectives from epistemology and philosophy of science*. Routledge, p 30
48. Durán JM (2019) Models, explanation, representation, and the philosophy of computer simulations. In: *Proceedings of the IACAP*, forthcoming
49. Krohs U (2008) How digital computer simulations explain real-world processes. *Int Stud Philos Sci* 22(3):277–292. <https://doi.org/10.1080/02698590802567324>
50. Cartwright N (1983) *How the laws of physics lie*. Clarendon Press
51. Mäki U (1994) Isolation, idealization and truth in economics. *Poznan Stud Philos Sci Hum* 38:147–168
52. Weirich P (2011) The explanatory power of models and simulations: a philosophical exploration. *Simul Gaming* 42(2):155–176. <https://doi.org/10.1177/1046878108319639>
53. Durán JM (2017) Varying the explanatory span: scientific explanation for computer simulations. *Int Stud Philos Sci* 31(1):27–45. <https://doi.org/10.1080/02698595.2017.1370929>
54. Humphreys P (2004) *Extending ourselves: computational science, empiricism, and scientific method*. Oxford University Press
55. Gelfert A (2018) Models in search of targets: exploratory modelling and the case of turing patterns. In: Christian A, Hommen D, Retzlaff N, Schurz G (eds), *Philosophy of science: between the natural sciences, the social sciences, and the humanities*. Springer International Publishing, pp 245–269. https://doi.org/10.1007/978-3-319-72577-2_14
56. Massimi M (2019) Two kinds of exploratory models. *Philos Sci* 86(5):869–881. <https://doi.org/10.1086/705494>
57. Winsberg E (2019) Computer simulations in science. In: Zalta EN (ed) *The Stanford encyclopedia of philosophy* (Winter 2019). Metaphysics Research Lab, Stanford University. <https://plato.stanford.edu/archives/win2019/entries/simulations-science/>
58. Barberousse A, Franceschelli S, Imbert C (2009) Computer simulations as experiments. *Synthese* 169(3):557–574. <https://doi.org/10.1007/s11229-008-9430-7>
59. Beisbart C (2018) Are computer simulations experiments? And if not, how are they related to each other? *Eur J Philos Sci* 8(2):171–204. <https://doi.org/10.1007/s13194-017-0181-5>
60. Jantzen BC (2016) Dynamical kinds and their discovery [Cs, Stat]. <http://arxiv.org/abs/1612.04933>
61. Hennig C (2015) What are the true clusters? *Pattern Recogn Lett* 64:53–62. <https://doi.org/10.1016/j.patrec.2015.04.009>
62. Ören TI (2011) A critical review of definitions and about 400 types of modeling and simulation. *SCS M&S Mag*, 2(3):142–151. <http://scs.org/wp-content/uploads/2016/12/2011-04-Issue06-6.pdf>. Accessed 02 Aug 2019
63. Experiment (n.d.) In *Cambridge’s dictionary*. Retrieved from <https://dictionary.cambridge.org/pt/dicionario/ingles/experiment>. Accessed 02 Aug 2019
64. Ören T, Mittal S, Durak U (2017) The evolution of simulation and its contribution to many disciplines. In: *Guide to simulation-based disciplines*. Springer, Cham, pp 3–24
65. de França BBN, Travassos GH (2016) Experimentation with dynamic simulation models in software engineering: planning and reporting guidelines. *Empir Softw Eng* 21(3):1302–1345
66. Rahmandad H, Sterman JD (2012) Reporting guidelines for simulation-based research in social sciences. *Syst Dyn Rev* 28(4):396–411
67. Alpaydin E (2016). *Machine learning: the new AI*. The MIT Press







68. Meraji S, Tropper C (2010) A machine learning approach for optimizing parallel logic simulation. In: Proceedings of 39th International conference on parallel processing, pp 545–554
69. Toma S (2014) Detection methodology and identify multiple faults in complex systems from discrete events and neural networks: applications for wind turbines. Thesis report, University of Corsica
70. Floyd MW, Wainer GA (2010) Creation of DEVS models using imitation learning. In: Proceedings of SCSC'10, San Diego, CA, USA, pp 334–341
71. Saadawi H, Wainer G, Pliego G (2016) DEVS execution acceleration with machine learning. In: Proceedings of TMS-DEVS'16, pp 16
72. Rachelson E, Quesnel G, Garcia F, Fabiani P (2008) A simulation-based approach for solving generalized semi-Markov decision processes. In: Proceedings of ECAI'08 conference. IOS Press, Amsterdam, The Netherlands, pp 583–587
73. Kessler C, Capocchi L, Santucci JF, Zeigler BP (2017) Hierarchical Markov decision process based on DEVS formalism. In: Proceedings of WinterSim'17, Dec. 3–6, 2017, Las Vegas, NV, USA, pp 1001–1012
74. Seo C, Zeigler BP, Kim D (2018) DEVS Markov modeling and simulation: formal definition and implementation. In: Proceedings of TMS '18, San Diego, CA, USA, Article 1, p 12
75. Puterman ML (1994) Markov decision processes: discrete stochastic dynamic programming, 1st edn. Wiley, New York, NY, USA
76. Arsham H (1998) Techniques for Monte Carlo optimizing. *Monte Carlo Methods Appl* 4:181–229
77. Sutton RS, Barto AG (1998) Introduction to reinforcement learning, 1st ed. MIT Press, Cambridge, MA, USA
78. Graciano Neto VV, Garcés L, Guessi M, Paes C, Manzano W, Oquendo F, Nakagawa E (2018a) ASAS: an approach to support simulation of smart systems. In: Proceedings of 51st Hawaii conference on systems sciences (HICSS). Big Island, Hawaii, USA, pp 5777–5786
79. Graciano Neto VV, Manzano W, Kassab M, Nakagawa EY (2018b) Model-based engineering & simulation of software-intensive systems-of-systems: experience report and lessons learned. In: Proceedings of the 12th European conference on software architecture: companion proceedings. ACM, Madrid, Spain (Article No. 27)
80. Graciano Neto VV, Paes CE, Rohling AJ, Manzano W, Nakagawa EY (2019) Modeling & simulation of software architectures of systems-of-systems: an industrial report on the Brazilian space system. In: 2019 Spring simulation conference (SpringSim). IEEE, pp 1–12
81. Oxford English Dictionary (2020) <http://oed.com>
82. Krishnan A (2009) What are academic disciplines? Some observations on the disciplinarity vs. interdisciplinarity debate. University of Southampton, NCRM Working Paper Series
83. Congress (2020) Library of Congress search: <https://www.loc.gov/books/?all=true&q=%22modeling+and+simulation%22>
84. Zeigler BP (1976) Theory of Modeling and Simulation. Academic Press
85. Ören T, Turnista C, Mittal S, Diallo S (2017a) Chapter 13: Simulation-based learning and education. In: Mittal S, Durak U, Ören T (eds) Guide to simulation-based disciplines: advancing our computational future. Springer
86. Wikipedia (2020) <https://en.wikipedia.org/wiki/Simulation>
87. Mittal S, Durak U, Ören T (eds) (2017) Guide to simulation-based disciplines: advancing our computational future. Springer.
88. Glotzer SC et al (2009) International assessment of research and development in simulation-based engineering and science. World Technology Evaluation Center, Baltimore, MD
89. Page E (2017) Chapter 2: Modeling and simulation (M&S) technology landscape. In: Mittal S, Durak U, Ören T (eds) Guide to simulation-based disciplines: advancing our computational future. Springer, pp 3–24

90. Valdivida WD, Clark BY (2015) The politics of Federal R&D: a punctuated equilibrium analysis. Brookings Institute
91. Industrial Research Institute (2016) 2016 Global R&D funding forecast. R&D Magazine
92. Forbes JR, Ortiz S (2007) H.Res. 487: Recognizing modeling and simulation as a National critical technology. Available from: https://www.scs.org/newsletters/2010-01/index_file/Files/MSResolution487.pdf
93. Tolk A, Ören T (eds) (2017) The profession of modeling and simulation: discipline, ethics, education, vocation, societies, and economics. Wiley
94. Zeigler BP, Praehofer H, Kim TG (2000) Theory of modeling and simulation: integrating discrete event and continuous complex dynamic systems, 2nd edn. Academic Press, London



History of Simulation

17

Bernard P. Zeigler , Breno Bernard Nicolau de França ,
Valdemar Vicente Graciano Neto , Raymond R. Hill ,
Lance E. Champagne , and Tuncer Ören 

Abstract

The chapter on history of simulation is foundational contribution to the SCS M&S Body of Knowledge that reviews the development of the discipline. The development of continuous and event-oriented simulation with the support of accompanying languages is the topic of one section. How simulation evolved to support experiments and experimentation is another facet of history. Finally, the evolution of simulation in analysis to support real-world decision making is evaluated in more detail. The chapter closes with a tabular view on previous studies on a M&S BoK, going back twenty years.

Keywords

Modeling and simulation · Body of Knowledge · Simulation languages

B. P. Zeigler (✉)
University of Arizona, Tuscon, US
e-mail: zeigler@rtsync.com

B. B. Nicolau de França
Universidade Estadual de Campinas, Campinas, Brazil
e-mail: breno@ic.unicamp.br

V. V. Graciano Neto
Federal University of Goiás, Goiania, Brazil

R. R. Hill · L. E. Champagne
Air Force Institute of Technology, Dayton, OH, USA
e-mail: lance.champagne@afit.edu

T. Ören
University of Ottawa, Ottawa, ON, Canada

17.1 General History of Modeling and Simulation

Bernard P. Zeigler

History provides a valuable perspective on the development of simulation as a model-based activity aimed at understanding and gaining control over reality (see the framework for modeling and simulation elucidated in Chap. 1). Perhaps, we are fortunate that reality displays multifarious regularities that can be exploited to construct useful models. The historical period of computer simulation as we know it spans less than a century (at this writing in 2019). However, some insight can be gleaned by regressing to the earliest references to human activity that foreshadow today's model-based activity.

17.1.1 “Modeling and Simulation” in Pre-History

Apparently, humans started counting at least 50,000 years ago [1]. Counting is a form of model-based activity which relies on the stability and persistence of common objects and their ability to be identified as members of a set of interest. The model abstraction underlying counting is the set of integers. Computation involves iterating through a one–one correspondence of the integers with the set of objects of interest. Tally marks on a tree trunk etched by an axe could provide the computational substrate needed to perform this operation. The abacus [2], dating as far back as 2700 BC, provides an analog form of decimal representation that enables efficient arithmetic calculation. It can be regarded as the earliest computing technology to support simulation of models of processes that are amenable to basic arithmetic manipulation. For example, consider using addition to predict the amount of wine in a jug to which the contents of another jug have been poured. Mathematical notation, a system of symbolic representations of mathematical objects and ideas, stems from 2000–1800 BC.

Throughout such history, calendar and navigation tools have been used in increasingly sophisticated forms [3]. They can also be considered as analog representations of time and space that enable predictions of recurring events and persistent locations in space (viz., some of the most important regularities mentioned above). Throughout the modern historical period, advances in representations in relatively permanent symbolic memorialization improved the written form. In the early eighteenth century, Leibnitz and Newton (in competitive fashion) developed the calculus, an abstract representation of motion, but Leibnitz recognized the need to provide a simulator to generate the behavior of models (of heavenly bodies and falling objects on earth): “Leibnitz actually designed and built a working calculating machine, the Stepped Reckoner ...inspired by the somewhat earlier work of Pascal, who built a machine that could add and subtract. Leibniz’s machine could add, subtract, divide, and multiply, and was apparently the first machine with all four arithmetic capabilities.” Since that time until the early 1950s continued development in computation technology was, in effect, devoted to

improving the simulation of differential (and difference) equation models expressed in Newton-Leibnitz notation cf., Babbage's Analytical Engine, hand calculators, and electronic analog computers. Interestingly, humans were referred to as "computers" and were organized into teams that computed ballistic missile trajectories in the Second World War before the advent of the electronic digital computer.

17.1.2 Continuous System Simulation

Continuous System Simulation Language (CSSL) was developed by a distinguished team at SCi (currently SCS) [97]. CSSL was a very successful and influential language at that time. System Dynamics (SD) is a modeling methodology developed since the late 1950s at the MIT School of Industrial Management by the engineer Jay Wright Forrester [4]. First expressed in the language DYNAMO, typical systems amenable to be described with SD are those for which complexity hinders the overall understanding by relying exclusively on domain-based first principles (e.g., demographic, biological, economic, environment, health, and managerial systems). Dynamics are mostly described by basic components of type "level" (or "stock") representing state variables, and "rate" representing first order differential equations. SD operates on a continuous-time, continuous-state scheme yielding a set of ordinary differential equations. DYNAMO tightly coupled together model specification with simulation technique. The modeler himself "programs" both his domain-specific knowledge along with the differential equation solver (1st order Euler method). Current environments supporting SD (e.g., Stella [5] provide a palette of global solvers to choose from, so the modeler focuses mainly on constructing the model visually, a feature present in current simulation environments.

17.1.3 Event-Oriented Simulation

The first use of a form of digital simulation roughly identified as event-oriented simulation appeared in the 1950s. An event-oriented simulation language provides a model representation that permits generation and analysis of system behavior. According to Nance [6], this provision entails six requirements: generation of random numbers, process transformation to statistical distributions, list processing capability, statistical analysis routines, report generation, and timing control. At its advent, it was mainly thought to be distinct from classical simulation in being a form of programming associated with the recent introduction of the digital computer and applied to operational research problems. In contrast, classical simulation was taken to be a form of numerical solution applicable to physics and related sciences whose speed could be greatly increased with mechanical rather than hand calculation. The distinctive trend of computational science is the continual trend toward greater abstraction and identification of the true underlying commonalities and distinctions between apparently different phenomena. This trend is realized in

the evolution of concepts relating to event-oriented simulation that transpired since its inception and which forms the core of the historical perspective on discrete event simulation.

Tocher [7] first conceived of discrete events *as* the right abstraction to characterize the models underlying the techniques that he and others were adopting in the mid-1950s. According to Hollocks [8], Tocher's core idea was of a system consisting of individual components, or "machines," progressing as time unfolds through "states" *that change only at discrete "events."* The history of discrete event simulation (now using Tocher's characterization) began with Monte Carlo sampling and the recognition of the limitations of analytic queueing analysis. These forerunners lead to machine language coding, to different attempts to handle specific real systems, and, eventually to general-purpose tools. In his history of discrete event simulation programming languages, Nance [6] breaks this history until 1986 into four periods: The Period of Search (1955–1960), The Advent (1961–1965), the Formative Period (1966–1970), the Expansion Period (1971–1978), and Consolidation and Regeneration (1979–1986).

17.1.4 Discrete Event Simulation Programming Languages

Early simulation programming languages were developed in the USA, including GSP [9], later GPSS [10], GASP [11], and SIMSCRIPT [12] in the USA; SIMULA in Norway [13]; and in the UK, GSP [7] (not the same as Gordon's GSP) and CSP [14]. The modeling strategies behind these programming languages became encapsulated in the concept of world views or conceptual frameworks.

The four major conceptual frameworks for DES emerged: (i) event scheduling, (ii) activity scanning, (iii) three-phase approach, and (iv) process interaction. These "world views" guide scientists in the design and the development of simulation models [15]. Among these frameworks, activity scanning is also called two-phase approach, the first phase being dedicated to simulation time management, the second phase to the execution of conditional activities (e.g., during scanning, execution of simulation functions depends on the fulfillment of specific conditions). The three-phase approach is an optimization of the activity scanning approach. This optimization is interesting for systems in which potential activities can be detected at each time step. The first phase is the same as in the activity scanning approach. The second phase is different since it handles the execution of all unconditional activities (avoiding rules scanning for rules known to always to be fired). The third phase is then similar to the second phase of regular activity scanning (an activity is considered and executed if the corresponding rule can be fired). These are also familiar as rule-based programming (expert systems) [16]. For example, in the Control and Simulation Language (CSL) when a rule is "fired" a corresponding action is taken and the system state is updated [14]. This approach is often considered to be dual with the event-scheduling method. A (partial) list of today's discrete event academic and commercial simulation software is available in Wikipedia DES [17].

17.1.5 Model-Based Simulation

The concept that the model underlying a simulation program should be separated from other elements of a simulation study (see Chap. 1 of BoKMS) first appeared in “Concepts for Advanced Simulation Methodologies” [18]. The article pointed out the shortcomings of existing simulation software and offered the then novel concepts to develop advanced simulation environments. The term “model-based” simulation [19, 20] aims to lay down the foundations for advanced methodology of simulation emphasizing the importance of central role of the models in simulation and opens up the possibility of novel methods for model processing. An account of the early development of the model-based simulation approach and the intellectual resistance it encountered is provided in Refs. [21, 22].

Ören [23, 24] developed the first simulation model specification language called GEST (General System Theory Implementer) based on Wymore’s book (“A Mathematical Theory of Systems Engineering” [25]). It concentrated on systems represented mostly by ordinary differential equations, aimed to ease robust specification and readability, and aimed at automatic translation to generate a simulation program in a general-purpose programming language [26, 27].

As the first system theory-based modeling and simulation language, GEST aimed to provide a language faithful to Wymore’s system theory that was well defined as understood in computer science (i.e., has well-defined syntax (via BNF) and semantics). GEST supported hierarchical (or nested) composition implementing Wymore’s coupling recipe concept for systems represented mostly by ordinary differential equations (later included finite state systems). The underlying goal was to “reduce” Wymore’s mathematical formalism to computational form so that modelers could take advantage of the digital computer while enjoying the benefits of the theory.

Another approach to such “reduction to practice” was developed by Zeigler a few years later and took off from the system specification theory exposed in his 1976 book [28]. Zeigler defined such set-theoretic specifications as Differential Equation System Specification (DESS), Discrete Time System Specification (DTSS), and Discrete Event System Specification (DEVS) as shorthand means of specifying Wymore’s general system objects. For an introduction to DEVS, see, for example, Wikipedia DEVS [29]. DEVS took Tocher’s discrete event idea one step further using the set theory of logicians and mathematicians [30, 31] and its use by Wymore [25].

As with GEST, the goal behind the DEVS approach was to provide a computational basis that was faithful to Wymore’s concepts, but the difference lay in seeking an iterative form for such specifications. You can understand this requirement by recognizing that Wymore’s system definition (given above) essentially defines the global solution of, say, a differential equation without providing a means to generate this solution. The most natural means of generating a system trajectory on the digital computer is by iteration or marching forward in time, either in fixed steps or by picking off imminent events from a calendar. The latter two forms of iteration, time step-based and event-driven, were formalized in the DTSS and DEVS formalisms and shown to generate the expected subclasses of

Wymore's systems. The subsequent concentration on DEVS stemmed from the recognition that DEVS was the more general formalism. The reason for this is that DEVS includes DTSS within it, i.e., any system generated by a DTSS can be generated by a DEVS. Later the expressiveness of DEVS was shown to include differential equation systems through the characterization of their time-stepped and quantized solvers [32].

Around the time of the emergence of DEVS as the computational basis for systems simulation, another important trend took hold. Object orientation (OO) was first introduced in simulation [33] and later spread to programming in general. It is fair to say that OO is at the heart of current information technology, so much so, that its presence is often taken as a given. For simulation modeling, DEVS and OO formed an ideal marriage. DEVS brought the formal and conceptual framework of dynamic systems while OO provided a rapidly evolving wealth of implementation platforms for DEVS models and simulations—first in languages such as Java and C++, and later in network and Web infrastructures, and today continuing in the future toward Cloud information technologies [34]. The first implementation of DEVS in object orientated form was described in Zeigler [35], and there are currently numerous such implementations, some of which are listed in Wainer DEVS Tools [36].

Zeigler [37] characterized these world views showing that they could all be represented as subclasses of DEVS. This representation also suggested DEVS's universality for discrete event models including those or other representations such as Timed Automata and Petri Nets. Zeigler et al. [32] later proved DEVS's ability to uniquely represent any system with discrete event input/output behavior. At the same time, a distinction between modular and non-modular DEVS was made, showing that the world views fit within the non-modular category. Moreover, while the modular class was shown to be equivalent to that of the non-modular one, it better supported the impending concepts of modularity, object orientation, and distributed processing on the software engineering horizon. Zeigler and Muzy [38] provide a timeline of DEVS research developments (see also Zeigler et al. [39]).

17.2 History of Simulation for Experimentation on Digital Computers

Breno Bernard Nicolau de França and Valdemar Vicente Graciano Neto

As mentioned in the very first chapter of this BoK, one of the main purposes of simulation is to perform experiments. Hence, understanding simulation as the act of “*performing goal-directed experiments with models of dynamic systems*” leads the simulation community to strive for more robust strategies for exploring the simulation models through appropriate experimental frames (see Sect. 1.4.2). Particularly, the concept of *experimental frame* supports the definition of a clearer scope for experiments by specifying the conditions under which the system is observed or experimented with. The source system is then intentionally abstracted, and its observation is reduced to a subset of the myriad of variables that could be observed [39, 40].

Over the years, simulations have certainly been applied for scientific purposes, particularly for proofs and prediction of properties, such as in climate or car crashes simulations. Simulation enables scientists to experiment with a model rather than the real-world object itself, which can be very useful for complex scenarios where analytical models are not enough or for scenarios where the real experiment is impossible or expensive. Thus, from a philosophical and epistemological point of view, simulations have the ability to support experiments and yield hypotheses and conclusions about source systems being simulated [41, 42].

In Statistics, the area addressing concerns of establishing experimental frames is called Design of Experiments (DOE) [43], which has a long history even before its application to computer simulation experiments. Kleijnen and van Groenendaal recall references from 1930, when Sir Ronald Fisher started developing experimental designs for agriculture experiments. In 1950, George Box further developed these techniques inspired by Fisher applications in chemistry. Since then, statistical techniques for simulation-based experiments have evolved and been widely studied theoretically and practically in several areas of knowledge.

Richard Conway [44] is credited with the beginning of experiments in computer simulation as a research area, initially called “analysis methodology” [45]. He described simulation experiments in three phases: (i) model implementation or development, (ii) strategic planning, and (iii) tactical planning. The first phase comprises the elaboration of the simulation model. Strategic planning (the second phase) stands for the design of a simulation experiment or establishing the experimental frame. Finally, tactical planning consists in the determination of how each simulation run in the experimental design should be performed. Conway concentrated his work on tactical planning, discussing issues on the estimation of steady-state parameters, variance-reduction techniques, and statistical approaches for comparisons of alternative solutions.

Since the early 1960s with Conway’s work, the research on how to conduct better simulation experiments has improved significantly, developing methodological support for relevant topics, such as design and analysis of simulation experiments [46], determination of appropriate batch sizes, output analysis [47], and others. More specifically, a core aspect regards the experimental designs for simulation experiments, which allow to compare alternatives, search for optimal solutions, sensitivity analysis, metamodeling, and validation [48]. In a historical perspective, simulation experiments were strongly influenced by classical experimental designs such as *one factor at a time*, *full factorial*, and *incomplete* or *fractional factorial*. These designs work well for comparing alternatives, but they can be very limited when considering a huge number of factors in an experimental frame. Furthermore, screening designs support the decision of more influential factors. Finally, optimization designs concentrate on performing simulation experiments to obtain optimal configurations.

During the 1970s, Kleijnen deepened in the statistical analysis of simulation-based experiments with his seminal works [49, 50] and presented guidelines including techniques for statistical analysis and experimental designs [51]. In this context, Response Surface Methodology (RSM) in simulation was first detailed in the monograph [49, 50]. However, RSM (unlike search heuristics) has not yet been

implemented widely into commercial-off-the-shelf simulation software. Regarding its application, Van den Bogaard and Kleijnen [52] reported one of the first case studies on RSM in random simulation, reporting on a computer center with two servers and three priority classes—with small, medium, and large jobs—estimating the 90% quantiles of the waiting times per class for different class limits, and applying RSM to find the optimal class limits [52].

In the same decade, Schruben and Margolin [53, 54] illustrate the adoption of two different variance-reduction techniques in a simulation-based experiment. They discuss the use of pseudo-random numbers. These authors bring in 1978 a retrospective of the 1970 decade, highlighting the seminal work of [49, 50] and also exposing how authors have agreed on the benefits brought by either statistically designed experiments and variance-reduction techniques.

In 1981, Ören presented a series of concepts and criteria to evaluate the acceptability and credibility of simulation studies [55]. These relate to data, model (conceptual and executable), experimental design, and methodology adopted to conduct the study. In the end of the decade, Sacks et al. [56] approached the pioneering work of Box and Draper [57] to discuss the design of physical experiments using simulation. They highlight the selection of inputs for a simulator as an experimental design problem and making predictions as a statistical problem that must consider stochastic models (or components) in computer experiments to quantify uncertainty.

In 1990, Balci presented guidelines for the success of simulation studies, organized according to the simulation model lifecycle and what the author calls “credibility assessment,” a set of V&V activities concerning each lifecycle step [58]. In 1992, Kleijnen published his important book on statistics for simulation [48]. Also in this period, considerable research has been conducted in Ranking & Selection procedures in general [59]; and specifically for simulation [60], which is now included into several commercial simulation products. Several researchers approached the use of variance-reduction techniques in the context of design of experiments and with R&S procedures for simulation experiments.

In the 2000 decade, Saltelli et al. published a book [61] on the sensitivity analysis for simulation-based studies, while other applications of simulation to specific domains started to be performed. Chen and colleagues [62] presented a literature review on statistical methods applied to computer experiments, more specifically, focusing on the goal of optimizing a complex system via deterministic simulation models. Since in previous decades researchers concentrated on suitable statistical methods to deal with simulations, in the end of 2000 decade, Kleijnen published a study on factor screening, i.e., searching for the most relevant factors (or inputs) to be varied when experimenting with real or simulated systems [63]. He published a review on Sequential Bifurcation (SB), which is a screening method for simulation experiments and summarized a case study, namely a supply-chain simulation with 92 factors where SB identified a shortlist with 10 factors after simulating only 19 combinations [63]. During the 2010 decade, Kleijnen also consolidated his work by publishing a second edition of his seminal book “Design and analysis of simulation experiments” [64].

Regarding the experimental concerns with simulation in other areas, in Medicine, Burton et al. present a checklist with relevant issues for developing simulation research protocols [65]. Generally, there is a concern with simulation model validity and with a statistically adequate experimental design. In Social Sciences, Rahmandad and Sterman published a set of reporting guidelines for simulation studies. Their proposal discusses three main aspects: model visualization for diagrams, model description for equations and algorithms, and simulation experiments design w.r.t. random numbers and optimization heuristics [66]. More recently, in 2016, de França and Travassos [67] presented several guidelines for planning and reporting simulation experiments in the context of software engineering.

In 2016, Nelson [68] revisited the Conway influential 1963 paper and stated tactical problems will still remain for digital simulation in the next 10 years (until 2025). Those problems include dealing with the analytics of large amounts of data generated by modern simulation, parallel simulation, and use of simulation to support decisions and how these factors impact on experiments feasibility, sensitivity, and optimization.

As a vast research field, simulation for experimentation has dense and solid research, both in theoretical and applied perspectives. Contributions across the years achieved a mature body of knowledge, establishing a more reliable path for conducting research supported by simulation experiments. However, challenges are still on the way as application areas demand more complex study settings and theoretical investigations as well.

17.3 Evolution of Computational Modeling and Simulation for Problem Solving

Raymond R. Hill and Lance E. Champagne

As Roberts and Pegden [69] note, simulation modeling is part of a problem-solving process. The focus of their paper is on the specific use of an executable model to support that problem-solving process. However, as noted in Hill and Miller [70] and Hill and Tolk [71], the use of simulation, in its most general form as a problem-solving paradigm, stretches back millennium in the military sciences; one might easily imagine this problem-solving paradigm just as applicable to non-military situations. Section 17.1 provides a great overview of the computational history associated with simulation as a general problem-solving framework. This chapter augments Sect. 17.1 by specifically examining computer-based modeling and simulation problem-solving.

17.3.1 Monte Carlo Modeling

Monte Carlo simulation is arguably the first real computerized instance of simulation. Goldsman et al. [72] indicate that the Monte Carlo method of using random

samples to conduct numerical calculations dates back to 1777 with the Buffon needle experiment. They also note the use of the method by Gosset in the early 1900s to develop the ubiquitous t-distribution. Complex mathematical calculations, when done manually, can be extremely time-consuming. For many years, this manual approach was the only option. Things started to change when electronic computing machinery started appearing in the 1940s as basically high-speed calculators. Scientists were quick to note these machines were adept at doing mathematical calculations quickly and accurately. One such calculation attainable with these new machines was the generation of pseudo-random numbers produced using an algorithm whose output (of those numbers) attained acceptable statistical properties. The Monte Carlo computational method of implementing statistical sampling for numerical calculations began in the late 1940s and truly demonstrated the power of these early computer systems. See the narrative by Metropolis [73] for a first-hand view of this key development in simulation. Extensions to Monte Carlo simulation continued to yield tremendous advances in numerical methods, risk analysis, non-dynamic simulation applications, and random variate generation.

17.3.2 Emergence of Simulation Modeling Languages

Early computers lacked much of the capability we now take for granted, such as internal memory. Each use of computers quickly outstripped the available memory in the machines. Thus, programs were written in machine or assembly code to save memory. These programming approaches were intellectually challenging and quite specific in their purpose. Fortunately, computer technology progressed quite rapidly and continues to progress, and along with that progression came the development of high-level programming languages as an easier means of producing executable computer programs. These high-level languages, which generally compiled into the necessary low-level code, provided more understandable forms of programming and started to appear in the late 1950s. FORTRAN appeared in 1954 and is widely regarded as the first popular general-purpose language [99]. These general-purpose languages, coupled with the ever-improving computer technology, enabled the development of specific simulations and general-purpose simulation infrastructures. See Nance [6] and Pegden [74] for more thorough discussions of computer simulation language development, and in particular, Barton et al. [75] for a discussion of how simulation has grown more efficient over the years.

17.3.3 The Arrival of Simulation Programs

Nance [6] lists the components of a simulation framework, or language as he describes it. Those components are:

- Random number generation;
- Random variate generation;

- List processing capabilities;
- Statistical analysis routines;
- Report generation; and
- A timing executive to control the simulation flow.

Nance [6], Robinson [76] as well as Goldsman et al. [72] credit Tocher with developing the first general-purpose simulator in 1960. The first widely popular simulation framework, the General Purpose System Simulator (GPSS), appeared in the 1960–1961 timeframe. Popular competing languages at the time include SIMULA and SIMSCRIPT, as well as a host of others discussed in Nance [6]. An excellent survey of how the simulation language domain grew in general is Pegden [74]. Early on, these languages were high-level programming languages. The simulation analyst needed to define the logic of the simulation and then develop the computer program corresponding to that logic using the high-level simulation language of choice. The simulation language provided key simulation infrastructure components, such as list processing routines, random variate generation routines, and some other common simulation characteristics, but it fell into the hands of the simulation analyst to put together the program necessary to faithfully represent the system or process of interest.

17.3.4 Expansion of Simulation

A simulation study will examine some aspect of a system or process of interest. Simulation problem-solving requires use of both conceptual and executable models. The conceptual model depicts the system or process of interest to include inputs, outputs, and interactions among components in the system or process. The executable model, referred to often as simply the simulation model, implements the conceptual model in the simulation language framework. The early simulation languages provided a high-level programming language for creating the latter. Support for the former was initially mostly left to the simulation analyst.

A conceptual modeling method “allows one to describe the time varying behavior” of the system(s) of interest such that they “can be analyzed using computer simulation” to generate subsequent analyses and insight [77]. As noted in Sect. 17.1.5, such methodologies began appearing in the late 1970s. Of such methodology, is IDEF2, an architectural methodology for dynamic system specification developed for the Air Force by the team of Pritsker and Associates and SofTech Inc. While focused on manufacturing systems, the IDEF2 methodology provided a graphical approach to define the problem of interest in such a way that it could then be examined with simulation. The graphical syntax defined in the IDEF2 methodology found its way into the very popular commercial product SLAM, developed by Pritsker and Associates. The benefit in SLAM was that the graphical conceptual model led directly to the simulation code required by the SLAM high-level simulation programming language. More important than the appearance of another simulation modeling framework was that SLAM was an early use of

specific graphical devices to define the conceptual model and then to build the executable model. This approach is now somewhat standard in the modern simulation modeling frameworks.

While the early graphical devices of the modeling method nicely linked to specific simulation modeling code, the process of laying out that conceptual model was still largely manual and paper-based. This changed as the computer technology dramatically shifted and the computer displays associated with the computing systems dramatically improved.

In the early 1980s, the personal computer emerged. With it came improved graphical processing and a personal linkage between the computing device and the user. No longer were computational jobs submitted to batch to run over night or required the user to interface with a mainframe computer via a terminal computing device. Instead, the personal computer allowed the simulation model developer to add their own general-purpose programming layer to the simulation language. Just as computer technology developed the general-purpose language over the machine-level code, the simulation model developers added the graphical user interface over the simulation language details. Simulation languages, such as SLAM and SIMAN, were coupled with outstanding, graphical-oriented programs yielding AWESIM and ARENA, respectively. While just a sample of two such programs, which is nowhere near comprehensive, these are representative of the now accepted approach for delivering a simulation framework. See Robinson [76] for more discussion on this aspect.

17.3.5 Simulation and Paradigm Shifts

Early simulations were quite limited. It took a couple decades for the simulation software to catch up with the promise of simulation as a problem-solving methodology. Not surprisingly, Harling [78] deemed simulation a method of last resort. But simulation use and influence did grow, spurred on by the rapid expansion of computing technology and the rich research agendas within simulation. Lucas et al. [79] declared it a method of first resort. This shift in attitude tracks well with the paradigm shifts due to simulation use and simulation technology.

Simulation-based optimization, or simply simulation optimization involves the use of simulation as an input evaluation method for some larger optimization process seeking a “best” combination of input values. For instance, Box and Hunter [80] introduced the response surface methodology (RSM). This methodology allows one to take a simulation meta-model and use it to guide the search for those input settings that produce better response output. RSM is an early simulation optimization approach. While the RSM process can be automated, it is a fairly human-intensive approach. Favored approaches now include heuristic search methods using the simulation to evaluate certain potential solutions. The tremendous power of simulation optimization is that the analyst has the descriptive power of the simulation model yet can attain the prescriptive modeling power previously only attainable via an optimization model. Fu and Henderson [81] provide a more thorough historical treatment of the methodology.

Robinson [76] recounts some of the early simulation frameworks to add visual tools. The visual representation of the implemented conceptual model provides a powerful mechanism for validating the model as well as achieving buy-in from the analytical customer that the simulation is correctly representing their system (or process). As the use of computer simulation grew, its influence on decision-making grew. Leadership confidence in the computer simulation results needed a firm basis beyond just the visual-based affirmation from the computer simulation graphics. Simulation model verification and validation grew in response to this need to build leadership confidence in simulation model-based analysis. Verification involves ensuring the executable model is built correctly while validation ensures that the simulation model conceived does in fact accurately mimic the system or process of interest. Today, simulation verification and validation is an assumed aspect of any simulation study. Sargent and Balci [82] provide a detailed history of simulation verification and validation.

Combat games are not new, and it can be argued that these were among the first simulations used. Loper and Turnitsa [83] and Hill and Miller [70] recount the pre-computer history of these combat games or simulations. Engineering computations, manually accomplished before the computer, were natural candidates for computational implementation. Quite rapidly, war game adjudication was computerized as well, which, in turn, led to the development of a variety of combat-focused, analytical, or constructive models. Battilega and Grange [84] provide a comprehensive survey of these models up until the 1980 timeframe. These combat simulations addressed combat in ground, naval, and air operations and varied levels of focus, or fidelity, from component performance, through system-versus-system encounters, to models of full theaters of war. This family of models, described in Hill et al. [85] as well as Hill and Miller [70] drastically changed the very nature of military operational research and continue to represent a significant aspect of military analysis.

Simulation for training has achieved great results due to computer simulation. The first such instances of trainers were realized in the early 1950s [83]. Advances in the visualization of the executing simulation led to the use of computer games for training and immersive gaming in the 1970s. With the growth of the Internet, also starting in the 1970s, the use of simulations across distributed platforms was envisioned and researched. By the 1990s, distributed simulation was really taking hold, with the growth of standards to promote simulation interconnectivity and architectures to connect a variety of systems. See Loper and Turnitsa [83] for a discussion of these developments.

A current focus is called Live, Virtual, and Constructive (LVC) simulations. Live denotes actual systems communicating with actual participants and various models and simulations. Virtual denotes actual participants involved with simulators (versus actual) systems, which are in turn communicating with the Live and Constructive components. The Constructive denotes the simulation models that are entirely contained with the computer, which are also communicating with the Live and Virtual components. The LVC provides enhanced training experiences as well as the ability to test and demonstrate large, complex system-of-systems

performance, testing that is largely impossible to attain in the traditional modes of system testing. See Hill et al. [86] and Hodson and Hill [87] for further discussion of LVC.

A final paradigm shift due to the growth of simulation technology discussed here is called the digital twin. A simulation represents a system or process of interest and produces results that should mimic results from the real system or those results that would come from that real system under similar configurations as used in the simulation. Why not let the simulation act as the real system then? This is the concept of the digital twin. Use the simulation to build and test the system (or process) of interest before the system (or process) is physically built or tested. Use the simulation to anticipate behaviors or situations in the actual system, before they actually occur in the real system. The digital twin concept could lead to fully digital system design, could refine use of autonomous vehicles, can help improve health care, and could be used to full train personnel on systems (or processes) before they actually see the system (or process), to name just a few applications. See Shaw and Fruhlinger [88] for an introduction and background on the digital twin concept.

17.3.6 The Simulation Research that Advanced the Simulation Use

Each of the components of the simulation framework has benefitted from the amazing research focused on improving those components of simulation. Early random number generation involved looking up values in tables. These were replaced with computational routines that were theoretically as well as practically limited. Modern uniform random number generators produce a practically unlimited supply of random numbers, with excellent statistical properties. See L'Ecuyer [89] for this history of this research.

Simulations require random instances from various probability distributions. Random variate generation routines convert the uniform random numbers to adhere to the distributions of interest. These routines are critical to producing accurate simulations. Kuhl [90] recounts the history of random variate generation while Cheng [91] recounts the advances made in correctly modeling the inputs to the simulations.

Simulation output is only useful if it can be used to make meaningful changes and promote leadership decision-making. The output of a simulation needs to be accurate, unbiased, and interpretable. The research in simulation output analysis has ensured the viability of simulation as a problem-solving methodology. Alexopoulos and Kelton [92] recount the advances in output analysis.

17.3.7 Simulation as the Method of Choice

Computer-based simulation has grown from a numerical calculation methodology to a ubiquitous presence. The engineering communities use simulation to analyze

and improve design concepts. The healthcare communities use simulation to improve all aspects of their operations (see Brailsford et al. [93] for the five-decade history). Manufacturing has become reliant on simulation to define and improve everything from manufacturing processes, through their warehousing operations, onto their supply-chain operations (see McGinnis and Rose [94] for a perspective on the history of simulation in manufacturing). Entertainment uses simulation routinely, from the computer-generated images that help to create the massively entertaining movies of today to the games and educational programs used by gamers of all ages and students.

Simulation will continue to grow. The research of the past is combining with the technology of the present and the future to bring on simulation capabilities that we have probably yet to imagine but will absolutely embrace and use to the benefit of all involved.

Disclaimer The views expressed in this article are those of the authors and do not reflect the official policy or position of the United States Air Force, the Department of Defense, or the United States Government.

17.4 Previous Studies on M&S BoK

Tuncer Ören

This section provides a selective literature survey in chronological order regarding the history of the M&S Body of Knowledge cumulating in this book.

2003

Birta, L. G. (2003). The Quest for the Modelling and Simulation Body of Knowledge. Proceedings. VIth Conference on Computer Simulation and Industry Applications, Instituto Tecnológico de Tijuana, Tijuana, Mexico, February 20–21, 2003, pp 7–13. (Keynote address). <https://www.site.uottawa.ca/~lbirta/pub2003-02-Mex.htm>

2005

Ören, T.I. (2005). Toward the Body of Knowledge of Modeling and Simulation (M&SBOK). *Proceedings of the Interservice/Industry Training, Simulation Conference*, Paper 2025, pp. 1–19. <https://www.site.uottawa.ca/~oren/pubs-pres/2005/pub-0513-MSBOK-IITSEC.pdf>

2006

Ören, T.I. (2006). Body of Knowledge of Modeling and Simulation (M&SBOK): Pragmatic Aspects. EMSS 2006 - 2nd European Modeling and Simulation Symposium, ISBN: 84- 690–0726-2, pp. 327–336. 2006 October 4–6, Barcelona, Spain.

- September 22–October 23,
University of Barcelona (Universitat Autònoma de Barcelona)
Invited Doctoral lecture
Barcelona lecture 1: Modeling and simulation (M&S): A comprehensive view
and an introduction to the Body of Knowledge.
<https://www.site.uottawa.ca/~oren/pubs-pres/2006/BarcelonaLecture-1-MSBOK.pdf>

2007

Ören, T. and Waite, B. (2007). Need for and Structure of an M&S Body of Knowledge. Paper presented at the Interservice/Industry Training, Simulation Conference (I/ITSEC), Orlando, FL. (presentation pdf <https://www.site.uottawa.ca/~oren/pubs-pres/2007/pres-IITSEC-MSBOK.pdf>).

2008

M&SCO. (2008). Department of Defense Modeling and Simulation Body of Knowledge (BOK). https://www.acqnotes.com/Attachments/DoD%20M&S%20Book%20of%20Knowledge.pdf?_ga=2.63595241.778764615.1622644614-1317686501.1622480173

2009

Waite, B. and T. Ören (2009). Modeling & Simulation Body-of-Knowledge Index (M&S BOK Index). Introductory Webinar. Prepared for: The SimSummit Round Table's M&S BOK Index Program, Senior Advisory Board.

2010

Lacy, L.W., Gross, D.C., Ören, T., and B. Waite (2010). A Realistic Roadmap for Developing a Modeling and Simulation Body of Knowledge Index. Proceedings of SISO (Simulation Interoperability Standards Organization) Fall SIW (Simulation Interoperability Workshop) Conference, Orlando, FL, September 20–24, 2010.

Ören, T.I. and B. Waite (2010). Modeling and Simulation Body of Knowledge Index: An Invitation for the Final Phases of its Preparation. SCS M&S Magazine, 1:4 (Oct). https://www.researchgate.net/publication/228749480_Modeling_and_Simulation_Body_of_Knowledge_Index_An_Invitation_for_the_Final_Phases_of_its_Preparation

Tolk, A. (2010). M&S Body of Knowledge: Progress Report and Look Ahead. SCS M&S Magazine, vol. 4, nb. 4, pp. 1–5. https://www.researchgate.net/publication/225038103_MS_Body_of_Knowledge_Progress_Report_and_Look_Ahead

- Workshop on Modeling and Simulation Body of Knowledge (M&S BoK) within the Summer Computer Simulation Conference of SummerSim Ottawa, Ontario, Canada, July 11–15, 2010. Organizers: Bill Waite and Tuncer Ören. Panelists: David Gross, Andreas Tolk, William Tucker, and John Williams. https://www.site.uottawa.ca/~oren/pubs-pres/2010/workshop_M&SBOK_notice.pdf

2011

Ören, T.I. (2011). A Basis for a Modeling and Simulation Body of Knowledge Index: Professionalism, Stakeholders, Big Picture, and Other BoKs. *SCS M&S Magazine*, 2:1 (Jan.), pp. 40–48. <https://scs.org/wp-content/uploads/2016/12/2011-01-Issue05-9.pdf>

2011, China

Invited Seminar: China lecture—1b: Modeling and Simulation: Body of Knowledge

<https://www.site.uottawa.ca/~oren/y/2011/09-MSBOK.pdf>

September 7-15, Beijing, China

Beijing University of Aeronautics and Astronautics; (aka) Beihang University School of Automation Science & Electrical Engineering

September 16-22, Changsha, China

National University of Defense Technology, System Simulation Lab

2012

Ören, T.I. (2012-Invited Keynote Paper). The Richness of Modeling and Simulation and its Body of Knowledge. Proceedings of SIMULTECH 2012, 2nd International Conference on Simulation and Modeling Methodologies, Technologies and Applications. Rome, Italy, July 28–31, 2012.

<https://www.site.uottawa.ca/~oren/y/2012/-08-MS-BOK.pdf>

(presentation: <https://www.site.uottawa.ca/~oren/y/2012/-08-MS-BOK-pres.pdf>).

(video presentation: <https://vimeo.com/48289950>).

2014

Ören, T.I. (2014-Invited paper). The Richness of Modeling and Simulation and an Index of its Body of Knowledge. (A revised and extended version of the keynote presentation at SIMULTECH'12, Rome, Italy, July 28–31, 2012). In: M.S. Obaidat, J. Filipe, J. Kacprzyk and N. Pina (eds) *Simulation and Modeling Methodologies, Technologies and Applications, Advances in Intelligent Systems and Computing*, Vol. 256, Springer, pp. 3–24. <https://download.e-bookshelf.de/download/0003/9234/61/L-G-0003923461-0013262512.pdf>

2015

Mustafee, N., Katsaliaki, K. and Fishwick, P. (2015). “A Review of Extant M&S Literature through Journal Profiling and Co-Citation Analysis” (M&S Body of Knowledge and Comprehensive and Integrative View). In: *Yilmaz L. (Ed.) Concepts and Methodologies for Modeling and Simulation: A Tribute to Tuncer Ören*. Springer Series in Simulation Foundations, Methods, and Applications Series. pp. 323–345. Springer. https://doi.org/10.1007/978-3-319-15096-3_15.

2017

Durak, U., T. Ören, and A. Tolk (2017). An Index to the Body of Knowledge of Simulation Systems Engineering (Chap. 2 of: Tolk, A. and T. Ören (eds.). The Profession of Modeling and Simulation. Wiley.

Ören, T. (2017). Modeling and Simulation Body of Knowledge (MSBOK) – Index. <https://www.site.uottawa.ca/~oren/MSBOK/MSBOK-index.pdf>

SCS Modeling and Simulation Body of knowledge (M&S BoK) – Index. <https://scs.org/body-of-knowledge-archive/>

2019

Lord, J.W. (2019). The Profession of Modeling and Simulations: Unifying the Organization. Master of Science Thesis in Modeling and Simulation in the College of Engineering and Computer Science at the University of Central Florida, Orlando, Florida. <https://stars.library.ucf.edu/cgi/viewcontent.cgi?article=7341&context=etd>

References

1. Wikipedia Count. Counting. <https://en.wikipedia.org/wiki/Counting>
2. Wikipedia Abacus. Abacus. <https://en.wikipedia.org/wiki/Abacus>
3. Wikipedia Timeline. Timeline of computing hardware before 1950. https://en.wikipedia.org/wiki/Timeline_of_computing_hardware_before_1950
4. Forrester JW (1961) Industrial dynamics. Pegasus Communications, Waltham, MA, p 464
5. Wikipedia STELLA. Stella (Programming Language). [https://en.wikipedia.org/wiki/STELLA_\(programming_language\)](https://en.wikipedia.org/wiki/STELLA_(programming_language))
6. Nance RE (1996) A history of discrete-event simulation programming languages. In: Bergin TJ, Gibson RJ (eds) History of programming languages, vol. II. ACM Press and Addison Wesley Publishing Company, New York, pp 369–427
7. Tocher KD (1963) The art of simulation. English Universities Press, London
8. Hollocks BW (2008) Intelligence, innovation and integrity-KD Tocher and the dawn of simulation. *J Simul* 2(3):128–137
9. Gordon G (1961) A general purpose systems simulator. In: Proceedings of EJCC, Washington D.C. Macmillan, New York, pp 87–104
10. Schriber T (1974) Simulation using GPSS. Wiley. ISBN 9780471763109
11. Kiviat PJ (1963) GASP—A general activity simulation program. Applied Research Laboratory, US Steel Corporation, Monroeville, PA
12. Markowitz HM, Hausner B, Karr HW (1962) Simscript: the simulation programming language. Rand Corporation Report, Rm-3310, Cambridge, MA
13. Nygaard K, Dahl O (1978) The development of the SIMULA languages. *ACM SIGPLAN Notices* 13:245–272
14. Buxton JN, Laski JG (1969) Control and simulation language. *Comput J* 5:194–199
15. Balci O (1988) The implementation of four conceptual frameworks for simulation modeling in high-level languages. In: Proceedings of the 20th winter simulation conference. San Diego, CA, pp 287–295
16. Chesnevar I, Maguitman A, Prescott Loui R (2000) Logical models of argument. *ACM Comput Surv* 32(4):337–383
17. Wikipedia DES. List of discrete event simulation software. https://en.wikipedia.org/wiki/List_of_discrete_event_simulation_software

18. Ören TI, Zeigler BP (1979) Concepts for advanced simulation methodologies. *Simulation* 32(3):69–82. SAGE Journals Online (<http://sim.sagepub.com/cgi/content/abstract/32/3/69>) (One of the **50 Most-Frequently Cited Articles** in *SIMULATION* (<http://sim.sagepub.com/reports/mfc1.dtl>) (6/50) as of March 1, 2010—updated monthly) Rankings are based on citations to articles on *SIMULATION* journal site from articles in HighWire-hosted journals (<http://highwire.org/lists/allsites.dtl#A>)
19. Ören TI (1984) Model-based activities: a paradigm shift. In: Ören TI, Zeigler BP, Elzas MS (eds) *Simulation and model-based methodologies: an integrative view*. Springer-Verlag, Heidelberg, Germany, pp 3–40
20. Ören TI, Zeigler BP, Elzas MS (eds) (1984) *Simulation and model-based methodologies: an integrative view*. NATO ASI series, Springer-Verlag, Berlin, Heidelberg, New York, Tokyo, p 651. http://www.site.uottawa.ca/~oren/pubs-pres/1984/pub-1984-03_GEST_NATO_ASI.pdf
21. Ören TI, Zeigler BP (2012) System theoretic foundations of modeling and simulation: a historic perspective and the legacy of a Wayne Wymore. *SIMULATION: Trans Soc Model Simul Int* 88
22. Zeigler BP, (2003) Autobiographical retrospectives: the evolution of theory of modeling and simulation. *Int J Gen Sys* 32(3):221–236
23. Ören TI (1971a) GEST: general system theory implementor, a combined digital simulation language. Ph.D. Dissertation, p 265. University of Arizona, Tucson, AZ. ACM Portal. <http://sabio.library.arizona.edu/record=b3042652~S9>
24. Ören TI (1984) GEST—a modelling and simulation language based on system theoretic concepts. In: Ören TI, Zeigler BP, Elzas MS (eds) *Simulation and model-based methodologies: an integrative view*. Springer-Verlag, Heidelberg, Germany, pp 281–335
25. Wymore AW (1967) *A mathematical theory of systems engineering: the elements*. Wiley, New York
26. Bleha LJ (1977) An implementation of the discrete portion of the combined digital simulation language GEST. Master's Thesis, Department of Electrical Engineering, University of Ottawa, Ottawa, Ontario, Canada
27. Dogbey FKA (1985) GEST Translator within the knowledge-based modeling system MAGEST. Master's Thesis, Computer Science Department, University of Ottawa, Ottawa, Ontario, Canada
28. Zeigler BP (1976) *Theory of modelling and simulation*. Wiley-Interscience, New York
29. Wikipedia DEVS. DEVS. <http://en.wikipedia.org/wiki/DEVS>
30. Whitehead AN, Russell B (1910) *Principia mathematica* 1 (1 ed). Cambridge University Press, Cambridge, JFM 41.0083.02
31. Wikipedia Bourbaki. Nicolas Boubaki. https://en.wikipedia.org/wiki/Nicolas_Bourbaki
32. Zeigler BP, Kim TG, Praehofer H (2000) *Theory of modeling and simulation: integrating discrete event and continuous complex dynamic systems*, 2nd edn. Academic Press, Boston
33. Birtwistle GM (1973) *SIMULA: its features and prospects*. Norsk Regnesentral, Oslo. Bongulielmi AP, Cellier FE (1984) On the usefulness of deterministic grammars for simulation languages. *ACM SIGSIM Simul Digest* 15(1):14–36
34. Wainer GA, Mosterman PJ (2009) *Discrete-event modeling and simulation: theory and applications*. Taylor & Francis
35. Zeigler BP (1987) Hierarchical, modular discrete event models in an object oriented environment. *Simul J* 49(5):219–230
36. Wainer G. DEVS Tools. <http://www.sce.carleton.ca/faculty/wainer/standard/tools.htm>
37. Zeigler BP (1984) Multifaceted modelling and discrete event simulation. Academic Press
38. Zeigler BP, Muzy A (2017) From discrete event simulation to discrete event specified systems (DEVS). In: 20th World congress of the international federation of automatic control (IFAC). Toulouse, France
39. Zeigler BP, Muzy A, Kofman E (2018) *Theory of modeling and simulation*, 3rd edn. Academic Press, Elsevier

40. Graciano Neto VV, Paes CEB, Rodriguez LMG, Guessi M, Manzano W, Oquendo F, Nakagawa EY (2017) Stimuli-SoS: a model-based approach to derive stimuli generators for simulations of systems-of-systems software architectures. *J Braz Comput Soc* 23(1):13–13:22
41. Grüne-Yanoff T, Weirich P (2010) The philosophy and epistemology of simulation: a review. *Simul Gaming* 41(1):20–50
42. Winsberg E (1999) Sanctioning models: the epistemology of simulation. *Sci Context* 12(2):275–292
43. Montgomery DC (2017). *Design and analysis of experiments*. Wiley
44. Conway RW (1963) Some tactical problems in digital simulation. *Manage Sci* 10(1):47–61
45. Nance RE, Sargent RG (2002) Perspectives on the evolution of simulation. *Oper Res* 50(1):161–172
46. Kleijnen JP, Sanchez SM, Lucas TW, Cioppa TM (2005) State-of-the-art review: a user's guide to the brave new world of designing simulation experiments. *INFORMS J Comput* 17(3):263–289
47. Alexopoulos C (2006) A comprehensive review of methods for simulation output analysis. In: *Proceedings of the 2006 winter simulation conference*. IEEE, pp 168–178
48. Kleijnen JP, van Groenendaal WJ (1992) *Simulation: a statistical perspective*. Wiley
49. Kleijnen JPC (1974) *Statistical techniques in simulation: Part I*. Marcel Dekker Inc., New York
50. Kleijnen JPC (1975) *Statistical techniques in simulation: Part II*. Marcel Dekker Inc., New York
51. Kleijnen JPC (1975) Statistical design and analysis of simulation experiments. *Informatie* 17(10):531–535
52. Van den Bogaard W, Kleijnen JPC (1977) Minimizing waiting times using priority classes: a case study in response surface methodology. (Ter discussie FEW; Vol. 77.056). Tilburg University
53. Schruben LW, Margolin BH (1978) Pseudorandom number assignment in statistically designed simulation and distribution sampling experiments. *J Am Stat Assoc* 73(363):504–520
54. Schruben LW, Coglianò VJ (1987) An experimental procedure for simulation response surface model identification. *Commun ACM* 30(8):716–730
55. Ören TI (1981) Concepts and criteria to assess acceptability of simulation studies: a frame of reference. *Simul Model Stat Comput* 24(4):180–189
56. Sacks J, Welch WJ, Mitchell TJ, Wynn HP (1989) Design and analysis of computer experiments. *Stat Sci* 409–423
57. Box GE, Draper NR (1959) A basis for the selection of a response surface design. *J Am Stat Assoc* 54(287):622–654
58. Balci O (1990) Guidelines for successful simulation studies. In: *Proceedings of winter simulation conference* (Dec. 9–12), pp 25–32
59. Bechhofer RE, Santner TJ, Goldsman D (1995) *Design and analysis of experiments for statistical selection, screening and multiple comparisons*. Wiley, New York
60. Goldsman D, Nelson BL (1998) Comparing systems via simulation. In: Banks J (ed) *The handbook of simulation*. Wiley, New York, pp 273–306
61. Saltelli A, Tarantola S, Campolongo F, Ratto M (2004) *Sensitivity analysis in practice: a guide to assessing scientific models*, vol 1. Wiley, New York
62. Chen VC, Tsui KL, Barton RR, Meckesheimer M (2006) A review on design, modeling and applications of computer experiments. *IIE Trans* 38(4):273–291
63. Kleijnen, J.P., 2009. Factor screening in simulation experiments: review of sequential bifurcation. In: *Advancing the frontiers of simulation*. Springer, Boston, MA, pp 153–167
64. Kleijnen JP (2015) Design and analysis of simulation experiments. In: *International workshop on simulation*. Springer, Cham, pp 3–22









65. Burton A, Altman DG, Royston P, Holder RL (2006) The design of simulation studies in medical statistics. *Stat Med* 25:4279–4292
66. Rahmandad H, Sterman JD (2012) Reporting guidelines for simulation-based research in social sciences. *Syst Dyn Rev* 28(4):396–411
67. de França BBN, Travassos GH (2016) Experimentation with dynamic simulation models in software engineering: planning and reporting guidelines. *Empiric Softw Eng* 21(3):1302–1345
68. Nelson BL (2016) ‘Some tactical problems in digital simulation’ for the next 10 years. *J Simul* 10(1):2–11. <https://doi.org/10.1057/jos.2015.22>
69. Roberts SD, Pegden D (2017) The history of simulation modeling. In: Chan WKV, D’Ambrogio A, Zacharewicz G, Mustafee N, Wainer G, Page E (eds) *Proceedings of the 2017 Winter simulation conference*. Institute of Electrical and Electronics Engineers Inc., Piscataway, New Jersey, pp 308–323
70. Hill RR, Miller JO (2017) A history of United States military simulation. In: Chan V, D’Ambrogio A, Zacharewicz G, Mustafee M, Wainer GA, Page E (eds) *Proceedings of the 2017 Winter simulation conference*. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, pp 346–364.
71. Hill RR, Tolk A (2017) A history of military computer simulation. In: Tolk A, Fowler J, Shao G, Yucesan E (eds) *Advances in modeling and simulation: seminal research from 50 years of winter simulation conferences*. Springer-Verlag, Berlin, Germany
72. Goldsman D, Nance RE, Wilson JR (2009) A brief history of simulation. In: Rossetti MD, Hill RR, Johansson B, Dunkin A, Ingalls RG (eds) *Proceedings of the 2009 Winter simulation conference*. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, pp 310–313
73. Metropolis N (1987) The beginning of the Monte Carlo method. *Los Alamos Sci Special Issue* 1987:125–130
74. Pegden CD (2017) Chapter 6: The evolution of simulation languages. In: Tolk A, Fowler J, Shao G, Yucesan E (eds) *Advances in modeling and simulation*. Springer International Publishing, Cham, Switzerland
75. Barton R, Nakayama MK, Schruben L (2017) History of improving statistical efficiency. In: Chan V, D’Ambrogio A, Zacharewicz G, Mustafee M, Wainer GA, Page E (eds) *Proceedings of the 2017 Winter simulation conference*. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, pp 158–180
76. Robinson S (2005) Discrete-event simulation: from the pioneers to the present, what next? *J Oper Res Soc* 56(6):619–629. <https://doi.org/10.1057/palgrave.jors.2601864>
77. SofTech Inc. (1981) *Integrated computer-aided manufacturing (ICAM) architecture Part II: Volume VI—dynamics model manual (IDEF2)*, AFWAL-TR-81-4023. DTIC ADB062459
78. Harling J (1958) Simulation techniques in operations research—a review. *Oper Res* 6(3):307–319
79. Lucas TW, Kelton WD, Sanchez PJ, Sanchez SM, Anderson BL (2015) Changing the paradigm: simulation, now a method of first resort. *Nav Res Logist* 62(4):293–303
80. Box GEP, Hunter JS (1957) Multi-factor experimental designs for exploring response surfaces. *Ann Math Stat* 28(1):195–241
81. Fu MC, Henderson SG (2017) History of seeking better solutions, aka simulation optimization. In: Chan V, D’Ambrogio A, Zacharewicz G, Mustafee M, Wainer GA, Page E (eds) *Proceedings of the 2017 Winter simulation conference*. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, pp 131–157
82. Sargent RG, Balci O (2017) History of verification and validation of simulation models. In: Chan V, D’Ambrogio A, Zacharewicz G, Mustafee M, Wainer GA, Page E (2017) *Proceedings of the 2017 Winter simulation conference*. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, pp 292–307

83. Loper M, Turnitsa C (2011) Chapter 16: History of combat modeling and distributed simulation. In: A. Tolk (ed) *Engineering principles of combat modeling and distributed simulation*. Wiley, New York, NY
84. Battilega JA, Grange JK (1984) *The military applications of modeling*. Air Force Institute of Technology, Wright-Patterson AFB, Ohio
85. Hill RR, McIntyre GA, Miller JO (2001) Applications of discrete event simulation modeling to military problems. In: Peters BA, Smith JS, Medeiros DJ, Rohrer MW (eds) *Proceedings of the 2001 Winter simulation conference*. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, pp 780–788
86. Hill RR, Tolk A, Hodson DD, Millar JR (2017) Open challenges in building combat simulation systems to support test, analysis and training. In: Rabe M, Juan AA, Mustafee N, Skoogh A, Jain S, Johansson B (eds) *Proceedings of the 2018 Winter simulation conference*. IEEE, Piscataway, NJ, pp 3730–3741
87. Hodson DD, Hill RR (2014) The art and science of live, virtual and constructive simulation for test and analysis. *J Defense Model Simul* 11(2):77–90
88. Shaw K, Fruhlinger J (2019) What is a digital twin and why it's important to IoT. *Networkworld*. <https://www.networkworld.com/article/3280225/what-is-digital-twin-technology-and-why-it-matters.html>. Accessed March 2021
89. L'Ecuyer P (2017) History of uniform random number generation. In: Chan V, D'Ambrogio A, Zacharewics G, Mustafee M, Wainer GA, Page E (eds) *Proceedings of the 2017 Winter simulation conference*. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, pp 202–230
90. Kuhl ME (2017) History of random variate generation. In: Chan V, D'Ambrogio A, Zacharewics G, Mustafee M, Wainer GA, Page E (eds) *Proceedings of the 2017 Winter simulation conference*. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, pp: 231–242
91. Cheng R (2017) History of input modeling. In: Chan V, D'Ambrogio A, Zacharewics G, Mustafee M, Wainer GA, Page E (eds) *Proceedings of the 2017 Winter simulation conference*. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, pp 181–201
92. Alexopoulos C, Kelton WD (2017) A concise history of simulation output analysis. In: Chan V, D'Ambrogio A, Zacharewics G, Mustafee M, Wainer GA, Page E (eds) *Proceedings of the 2017 Winter simulation conference*. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, pp. 115–130.
93. Brailsford SC, Carter HW, Jacobson SH (2017) Five decades of healthcare simulation. In: Chan V, D'Ambrogio A, Zacharewics G, Mustafee M, Wainer GA, Page E (eds) *Proceedings of the 2017 Winter simulation conference*. Institute of Electrical and Electronics Engineers, Inc., Piscataway, New Jersey, pp 365–384.
94. McGinnis LF, Rose O (2017) History and perspective of simulation in manufacturing. In: Chan V, D'Ambrogio A, Zacharewics G, Mustafee M, Wainer GA, Page E (eds) *Proceedings of the 2017 Winter simulation conference*. Institute of Electrical and Electronics Engineers, Inc, Piscataway, New Jersey, pp 385–397
95. Sci. The SCI continuous system simulation language (CSSL). *Simulation* 9(6):281–303
96. History of Programming Languages. Wikipedia. 2021. https://en.wikipedia.org/wiki/History_of_programming_languages



Core Research Areas

18

Paul Fishwick , Saikou Y. Diallo , Umut Durak , Baocun Hou, Bo Hu Li, Chunhui Su, Yanguang Wang, Lin Zhang , Xu Xie, Longfei Zhou , Bernard P. Zeigler , Thorsten Pawletta , Hendrik Folkerts, and Saurabh Mittal 

Abstract

The SCS M&S Body of Knowledge is a living concept, and core research areas are among those that will drive its progress. In this chapter, conceptual modeling constitutes the first topic, followed by the quest for model reuse. As stand-alone applications become increasingly rare, embedded simulation is of particular interest. In the era of big data, data-driven M&S gains more interest as well. Applying the M&S Framework (MSF) to enable neuromorphic architectures exemplifies the ability of simulation to meaningfully contribute to other fields as well. The chapter closes with sections on model behavior generation and the growth of simulation-based disciplines.

P. Fishwick (✉)
University of Texas at Dallas, Richardson, TX, USA
e-mail: Paul.Fishwick@utdallas.edu

S. Y. Diallo
Old Dominion University, Norfolk, VA, USA

U. Durak
German Aerospace Center, Cologne, Germany
e-mail: Umut.Durak@dlr.de

B. Hou
Midea Cloud Tech Co., Ltd, Beijing, China
e-mail: houbc2@meicloud.com

B. H. Li · L. Zhang
Beihang University, Beijing, China
e-mail: bohuli@moon.bjnet.edu.cn

L. Zhang
e-mail: johnlin9999@163.com

Keywords

Modeling and simulation · Conceptual modeling · Model reuse ·
Data driven M&S · Embedded M&S · M&S Framework (MSF) ·
Simulation-based disciplines

18.1 Conceptual Modeling

Paul Fishwick, Saikou Y. Diallo

We begin with discussing the basic ideas of conceptual modeling and then proceed to a description of how conceptual modeling is treated within the M&S community. We start with the general and then progress toward the specific.

Modeling is creating a simplified representation of something. This definition holds in virtually all disciplines from art, and the humanities, to mathematics, science, and engineering. If modeling is representation, then what does conceptual modeling mean? A definition of concept from the Oxford English Dictionary (OED) is “something conceived in the mind; a notion, idea, image, or thought.” Another more elaborate OED definition of concept is “a general idea or notion, a universal; a mental representation of the essential or typical properties of something, considered without regard to the peculiar properties of any specific instance or example.” From these two definitions, we are drawn to ideas and mental representations. Fishwick [1] presents a chapter on concept modeling, which is portrayed as a more informal, initial phase to model development.

C. Su · Y. Wang
CASICloud-Tech Co., Ltd., Beijing, China
e-mail: xiao_spring@163.com

Y. Wang
e-mail: wyg_850802@163.com

X. Xie
National University of Defense Technology, Changsha, China

L. Zhou
Duke University Medical Center, Durham, NC, USA

B. P. Zeigler
University of Arizona, Tucson, AZ, USA
e-mail: zeigler@rtsync.com

T. Pawletta · H. Folkerts
University of Applied Sciences in Wismar, Philipp-Müller-Str. 14, 23966 Wismar, Germany
e-mail: thorsten.pawletta@hs-wismar.de

S. Mittal
The MITRE Corporation, Dayton, OH, USA
e-mail: smittal@mitre.org

Mental modeling is present in psychology [2, 3] where a mental model is captured by some language or formalism. Some popular formalisms for this type of model are logic, concept maps, and mind maps. Logic [4] is a way to express natural language but in a more structured symbolic fashion. Mind maps [5] and concept maps [6] are similar. Both maps are directed graphs, represented visually by concepts, which are connected by relationships. The mind map centers on one primary concept and then branches out to sub-concepts. Mind maps contain wide latitude on drawing style and aesthetic choice. Concept maps have no central node and represent a network of concepts.

If we take a step back from mind maps and concept maps, we arrive at two more fundamental human activities: drawing and writing. Drawing and writing represent languages for describing and notating concepts. The statement “the oak tree has many thick, gnarly, branches which tend to twist and turn” is a kind of model since the language statement is a simplified representation of the tree. A drawing of the oak tree captures the tree but in a more convenient package: the drawing.

That drawing and writing define modeling at its most natural state, we can situate modeling as a fundamental, behavioral aspect of our species. We have been representing for a very long time. The notation that drawing and writing capture modeling conceptually also feeds into the idea that these creative practices come to fruition in our early education. We all learn verbal and written language and drawing at a young age and thus begin our conceptual modeling adventure.

The modeling and simulation community is at its root concerned with engineering models that combine mathematical and logical aspects of a problem. As a result, conceptual modeling tended to be subsumed with the natural abstractions provided by mathematics and logic. Conceptual models were therefore incorporated into design specifications, algorithms, and blueprints and were directly implemented into the expression of the simulation model in the language of the simulator in which it was executed. While that approach married well with the problem domains associated with engineering models, it did not equip engineers to deal with situations where specifying the problem was a necessary first step to solving it. Over the years, three major advances lead to the emergence of conceptual modeling as a major component of M&S.

First, the need to reuse models in order to answer new questions pushed M&S professionals to develop framework to make simulations interoperable. In that process, they soon discovered that they were more likely to build a “Frankenstein” solution made of disparate, incoherent models that they were to achieve a cohesive whole. They realized that rather than simply making simulation interoperate, they had to compose models. The desire to compose models revealed the limitations of using design specifications algorithms to capture conceptualizations. While these documentations provide what was done, it did not necessarily describe the intent of the model and the rationale for making design decisions. As a result, it was nearly impossible to determine the intersection of models and decide on which components of a model should be included in a desired solution. The struggles to achieve meaningful interoperability and composability pushed the M&S engineering community toward a separate conceptual modeling process that would help increase

model transparency and provide a solid basis for integration, interoperation, and composition of M&S solutions [7].

Second the emergence of computational social sciences [8] over the last two decades has given a renewed emphasis to the ability to first design a problem through a conceptualization process and then attempt to investigate it through simulation. In social sciences, the source of the model or the referent is a social system often described in the form of theories with varying levels of complexity and ambiguity. The ability to (1) faithfully represent theories and (2) abstract them to their bare axiomatic structures requires the development of very elaborate conceptual models. Conceptual models thus developed serve communication devices between the modelers and an expert audience to evaluate how well the modelers are adhering to the theory. As the modelers cease to be the experts, conceptual modeling became the means by which modelers capture knowledge from subject matter experts for the purpose of using simulation to answer a modeling question. Over time, conceptual modeling played a critical translator role between the organized expert knowledge captured in theories and the organized semi-formal knowledge required to answer a modeling question. Recently, some experts have argued that expert knowledge is biased and that it is important to understand and capture all worldviews around a situation [9]. They argue that a reference model that captures what is known and assumed about the model should first be specified and that the conceptual model should be a representative subset of the reference model. This concern of bias and myopia in the representation and design of the problem to be solved through modeling and simulation has catapulted conceptual modeling in the domains of philosophy of science and systems science. It has also opened the door to the use of modeling and simulation to study complex societal problems which is the third major advance.

As global challenges continue to grow and become more intertwined, policy experts are seeking ways to gain insight into the mechanisms of a problem and the impact of potential solutions. Policy analysts dealing with pandemics, climate change, globalization, and forced migrations are asking for tools that can help them frame a problem space and help define the contours of a solution space. Modeling and Simulation experts in conjunction with economists, social scientists, humanists, and philosophers [10] are increasingly developing models that seek to explore problems of interest to decision-makers. Conceptual modeling serves as the common language that experts from multiple disciplines use to communicate knowledge, assumptions, and worldviews. Conceptual models capture the consensus (or lack thereof) on the problem and represent the problem specification as understood by the experts. The model is useful in communicating with policy experts and serves as the basis for evaluating the recommendations derived from simulation results. Conceptual modeling is very valuable in that:

1. it uncovers and makes explicit knowledge and bias;
2. it reveals beliefs about the nature of the situation of interest (climate change, child sex traffic, etc.) and its underlying root causes and mechanisms;
3. it allows policymakers to observe and understand how their views differ from those of the experts and other policy professionals; and
4. it serves as the axiomatic structure for key assumptions and their justification.

All three of these factors have made conceptual modeling a critical and unavoidable step of the M&S lifecycle.

As a result of the emergence of conceptual modeling as part of a wider elicitation process, conceptual modeling tools include a wide range of solutions. The language of conceptual models varies from the informal (documents, reports) to semi-formal (UML, SYSML, etc.) to formal (domain-specific languages). Mind mapping tools are used to capture knowledge and relationships in the form of taxonomies or ontologies. Semi-formal tools used in systems and software engineering are effective in providing specifications that can be transformed into computational or mathematical models. However, conceptual modeling is an iterative process that is imbued in every aspect of the M&S process and the different models simply capture the consensus of an aspect of the problem at a point in time.

In a recent expert panel on conceptual modeling [11], the need to make such models as discussed in this section accessible to machine was an additional topic. If artificial intelligence applications shall help to identify, reuse, and compose simulation solutions, being able to read and understand the underlying assumptions and constraints captured in the conceptual model becomes pivotal [11].

18.2 Model Reuse

Saikou Y. Diallo, Umut Durak

Reuse is a highly desirable property of models because of the perceived savings in the time and effort involved in researching, developing, and documenting new models and simulations. While reuse has many definitions (formal and informal), it is generally understood as the ability to use an existing model to answer a modeling question. One of the major references in simulation mode reuse is [12].

In order to better understand reuse, it is important to separate (1) the referent which is the phenomena being model, (2) the model which is a representation of the phenomena, and (3) a simulation which is the execution of a model. Referents by their very nature are not reusable. However, modelers can recognize that they are dealing with a referent that they or others have modeled before in order to answer a modeling question. The recognition of a similar referent is often the first step toward model reuse.

For a modeler, models are reusable depending on how closely they match the new modeling question and how relevant the underlying assumptions bounding the model remain acceptable. Models that are reusable usually capture mathematical abstractions that describe physical and chemical phenomena (speed, motion, heat transfer, etc.) and rely on commonly accepted theories, practices, and assumptions (Newtonian Laws for instance). For example, physics-based models are largely reused and act as de-facto standards in some modeling communities [13, 14]. That can be also be categorized as conceptual model reuse.

Simulation reuse is largely dependent on platform, language, input and outputs. In the case where the modeler wants to reuse the simulation as-is, the ability to provide acceptable inputs and the relevance of the outputs in terms of resolution and scope (types of attributes, units of measure, frequency of sampling) drive the utility of the simulation. Full model reuse is at the far end of the reuse spectrum where the frequency is rather low and complexity is rather high; the other hand being code scavenging where the frequency is high, and the complexity is quite low [15]. From code scavenging to full model reuse, there is function reuse and component reuse. Further classifications were also made such as programming-level, frameworks-level, design-level, and COTS-level reuse [16].

Model reuse has long been studied by the simulation community. One of the early examples is [17] where Reese and Wyatt trying to establish the link between software and simulation reuse. Since the early efforts, model reuse is driven by the enabling architectural approaches from software engineering. There have been efforts around distributed computing and component-based software engineering approaches as the enabler of simulation reuse, some of which are High-Level Architecture (HLA) [18, 19], DEVS [20, 21], and Functional Mock-up Interface (FMI) [22, 23]. The emergence of simulation as a service (SaaS) [24] later brought a good step toward increasing the reuse of simulation models. In the case where the modeler wants to integrate the simulation with other components, the platform and language of the simulation play a larger role. The use of micro-service architectures [25] as a part of service-oriented architecture design pattern will further facilitate the integration of simulation components into a larger whole.

Reuse is often associated with interoperability and composability [26] where interoperability is the ability to connect simulations to answer a modeling question and composability is the ability to connect models to answer a modeling question. Interoperability, composability, and reuse are often conflated, although they are different [27]. Reuse is ultimately about the ability to map referents and recognize the difference between an existing model of that referent and the desired model of that referent. A failure to adequately map those aspects will result in wasting resources and increased frustration.

At the enterprise level, reuse, interoperability, and composability must be planned and managed. To maximize reuse standards in documentation, development, and tools must be promoted and enforced. Design patterns that separate data and functions tend to promote the reuse of both. Layered approaches, such as the Levels of Conceptually Interoperability Model (LCIM) provide a similar structure for governance and implementation as well [28]. Effective documentation of knowledge claims and knowledge management strategies and tools that promote discovery and transparency are key enablers of reuse. Ontology-based approaches have been investigated to streamline simulation reuse [29, 30].

18.3 Embedded Simulation/Ubiquitous Simulation

Baocun Hou, Bo Hu Li, Chunhui Su, Yanguang Wang

18.3.1 Connotation

Embedded simulation/ubiquitous simulation is a kind of new modeling and simulation technology which integrates real-time computing, ubiquitous computing, Internet of Things (IoT), and cloud computing.

Embedded simulation/ubiquitous simulation is a brand-new simulation mode that combines simulated space with physical/logical space, which seamlessly integrates simulator hardware/software, communication hardware/software, various sensors, and simulated world. Currently, digital twin is an area of increasing focus for embedded simulation/ubiquitous simulation, especially when the simulated model is represented as digital twin.

The important significance of embedded simulation/ubiquitous simulation is introducing the simulation technology into the real system and embedding seamlessly into everything/world for achieving simulation everywhere across the world. The simulation services can be accessed transparently whenever and wherever possible under the embedded simulation/ubiquitous simulation mode.

18.3.2 Technology System

18.3.2.1 Modeling Theory and Methodology

Modeling theory and methodology mainly include the primary modeling theory of the simulation system and its high-performance real-time simulation solution algorithm (secondary modeling method) for embedded simulation/ubiquitous simulation.

18.3.2.2 Simulation System Theory and Technology

Simulation system theory and technology mainly include the relevant theory and technology of hardware support environment, software support environment for operating system nucleus, high-speed communication system, and resource management system [31].

Theory and technology of simulation system construction and operation include real-time service, virtualization service, dispatching management service, load balancing service and resource management service and simulation service migration technology, as well as the relevant theories and enabling technologies of building and operating the simulation system using real-time/ubiquitous simulation engine, solution tool, and algorithm library.

18.3.2.3 Simulation Application Engineering Theory and Technology

Simulation application engineering theory and technology mainly includes the simulation application engineering theories and technologies supporting the embedded simulation/ubiquitous simulation application, as well as model library, algorithm library and database in the fields of national economy, national welfare, people's livelihood, national security, and the relevant theories and technologies.

18.3.3 Key Technologies

18.3.3.1 Embedded Simulation/Ubiquitous Simulation Modeling Technology

Oriented for the embedded system, embedded simulation/ubiquitous simulation modeling technology can be classified into two categories general modeling technologies and specific modeling technologies [32]. General modeling technologies include mechanism modeling, identification modeling, object-oriented modeling, multi-view modeling, and data visualization modeling. Specific modeling technologies include agent-oriented modeling, ontology-oriented modeling, distributed inference-oriented network modeling, high-performance real-time simulation technology, inference/evolution /self-organization modeling for mobile communication.

18.3.3.2 Embedded Simulation/Ubiquitous Simulation System Architecture

Based on the Internet, the hierarchical, networked and service-oriented embedded simulation/ubiquitous simulation system architecture focuses on the integration of simulation and physical spaces, with the aim of accessing to the simulation services as needed anytime and anywhere and transparently, so as to conform to the new requirements in the simulation application field under the "user-centric" and "implicit invocation" application mode [33–37].

Embedded simulation/ubiquitous simulation system architecture includes portal layer, core service layer, middleware layer, resource layer, and the security system for ubiquitous simulation system throughout the aforesaid all layers. Figure 18.1 shows the embedded simulation/ubiquitous simulation system architecture.

1. Resource layer: Various resources dynamically dispatched and applied by the ubiquitous simulation system are provided in the form of services, including computing service, data service, storage service, software service, model service, RTI service, etc.
2. Middleware layer: It consists of context-aware middleware, service grid middleware, and intelligent agent middleware.
3. Core service layer: The core service layer is composed of operation support service, application development environment, simulation application support service, and simulation service/task description specifications.

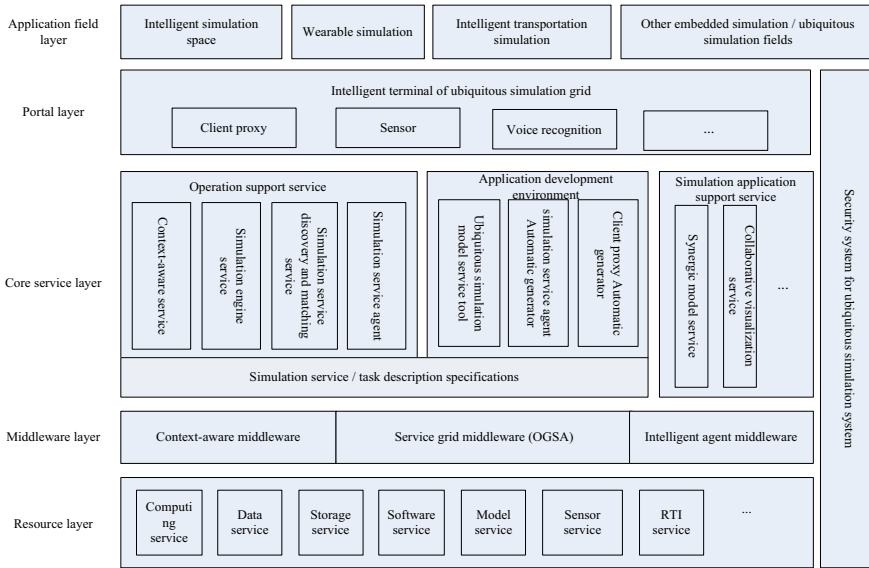


Fig. 18.1 Embedded simulation/ubiquitous simulation system architecture

4. Portal layer: The intelligent terminal of ubiquitous simulation system is vital to enabling the ubiquitous simulation application mode. Intelligent agent generated by the intelligent terminal of ubiquitous simulation system helps the user complete simulation service request, access, and simulation task execution.

18.3.3.3 Software Platform and Middleware of Embedded Simulation/Ubiquitous Simulation System

The software platform of embedded simulation/ubiquitous simulation system supports the deployment, operation and testing of embedded simulation application, including the framework of software platform, reusable component library, development kit, and open third-party application interface [37–40]. The middleware technology for embedded simulation/ubiquitous simulation system mainly includes such technologies as context awareness, service discovery, security control, and embedded-resource constrained device processing. The ubiquitous simulation middleware can be classified into data processing middleware, context-aware middleware, service discovery middleware, and integrated application middleware by functions, respectively. Through the functional integration of universal simulation middleware, comprehensive application services can be provided for users to improve system availability and usability.

18.3.3.4 Perception and Interaction Technologies Between Human and Embedded Simulation/Ubiquitous Simulation Service

The ultimate goal of the Ubiquitous computing mode is to integrate the information space consisting of communication and computer with the physical space where people live and work, support the user to acquire the personalized information service “anytime and anywhere” and transparently, so as to provide the user with an implicit interaction mode [34].

“User-centric” represents a significant change in the way of accessing to the simulation service, and a transformation of a simulator-centered application mode into a human-centered application mode. Computer is no longer the only access for user to acquire simulation service. The networking intelligent terminal will penetrate into people's living space by taking various forms, representing a carrier from which people acquire the simulation service, i.e., a ubiquitous simulation space.

Under the “user-centric” ubiquitous simulation application mode, all users will stay in a terminal environment with perceiving and networking functions, the intelligent terminal can automatically identify user's application requirements, then map the application requirements into the simulation ones, and the simulation tasks will be submitted to the ubiquitous simulation grid. The ubiquitous simulation grid searches for the required simulation services automatically, combines the simulation services, dynamically constructs the simulation application system, executes the simulation task, fulfill user's requirements, and returns the simulation result. A desired simulation result can be obtained as long as the user describes his application or simulation requirements based on a certain simulation language. Finally, a highly automated and intelligent simulation system is established.

18.3.3.5 Simulation Service Migration Technology for Embedded Simulation/Ubiquitous Simulation Mode

Simulation service migration technology for embedded simulation/ubiquitous simulation mode involves the definition of simulation service migration agent and its internal modules, simulation services and design for general interface associated with migration [41]. Depending on mobile agent and context-aware technologies, the functional and non-functional migration attributes of simulation service can be separated, ensuring the consistency of simulation service operation status, achieving the goal of the full life circle management of simulation service migration. This conforms to the application requirements of network bandwidth and resource configuration optimization, load balancing and support offline services in ubiquitous simulation.

18.3.3.6 Coordinated Management and Integration Technologies of Simulation Space and Physical/Logical Space

Ubiquitous simulation represents the integration of such technologies as cloud simulation, sensor network, mobile computing, context-aware computing, wearable computing, intelligent, agent and Semantic Web, aiming at enabling security

dynamic sharing, reuse and interoperability as well as collaborative scheduling operation and optimization of various simulation resources. Firstly, various software and hardware simulation resources are provided to serve people's life by virtue of cloud simulation technology, so as to "spread" the simulation resources ubiquitously. Secondly, simulation application terminal extends to the network everywhere based on its functions of perceiving, computing, and networking, which thoroughly breaks away from the bondage in time and space, conforms to the requirements of accessing simulation services "anytime and anywhere". Finally, simulation user, task, and service can be freely organized according to requirements, simulation service can be changed from "simulator-centric" to "user-centric."

18.3.3.7 System Security Technology

System security technology includes authentication, access control, privacy management, distribution control, and interoperability between mobile device and infrastructure, which provides security assurance to application services in a ubiquitous simulation environment [30, 42–44], such as middleware technology based on security rules, context-centered access control, public key-based or role-based authentication and intrusion detection, symmetric encryption algorithm, security group communication, and authentication. Currently, the existing system security technology calls for high-performance computing ability and storage capacity, the applicability on resource-constrained devices in ubiquitous simulation environment is somewhat inadequate. From an implementation perspective, focusing on the development of mobile terminal security control technology in a ubiquitous simulation environment will be an important research direction.

18.3.3.8 Embedded Simulation/Ubiquitous Simulation Application Technologies

The application technologies of embedded simulation/ubiquitous simulation include the construction technology of ubiquitous simulation solving environment, HLA/RTI network technique, resource service middleware, simulation resource service, semantic-based simulation model resource discovery, simulation resource service combination and system security mechanism and user management. The functions of simulation resource development, deployment/registration, and simulation resource retrieval/discovery are implemented in combination with the application scenarios of different industrial fields, leading to the ability to dynamically construct simulation application based on the service combination mode.

18.3.4 Development Tendency

18.3.4.1 Ubiquitous Simulation Service

Simulation resources are ubiquitous. Various software and hardware simulation resource services are provided to human life to shield the complicate and

heterogeneous ubiquitous simulation environment, in order to meet the requirements of invocation simulation services anytime, anywhere, and transparently [29].

18.3.4.2 Ternary Integration of “Human-Cyber-Physical”

With the development of cloud computing, IoT technologies, and the emerging heterogeneous terminals, ubiquitous simulation will develop the “human-centric” ubiquitous perception service and expand the perception and interoperability of existing equipment, which is a key supporting technology for developing the ternary integration of “human-cyber-physical.”

18.3.4.3 Ubiquitous Human–Computer Interaction

It integrates such advanced technologies as intelligent computing, sensor, and embedded system, which provides more intuitive and natural interactive experience in the simulation, physical and social spaces.

18.3.5 Application Scenarios

18.3.5.1 Intelligent Manufacturing

1. Simulation, control, and optimization in the manufacturing process

Embedded simulation is conducted on the control model of predictive control system for parameter matrix optimization of multiple-input multiple-output (MIMO) control system, and cooperative control simulation and optimization of the distributed system consisting of a plurality of controllers.

2. Simulation and dynamic configuration of production resources

Parameter estimation and rapid workshop scheduling optimization are enabled by building the desired prediction and production resource organization models for real-time workshop production simulation at the edge layer of intelligent manufacturing system.

3. Predictive maintenance

Predictive maintenance and fault detection process optimization are enabled by comparing various kinds of characteristic data generated by equipment health model and equipment during the production in the intelligent manufacturing system.

4. Intelligent logistics

By the means of embedded simulation, during the process of intelligent logistics, real-time simulation and optimization of work path can be carried out to realize intelligent dynamic path planning and improve logistics efficiency.

18.3.5.2 Intelligent Transportation

Massive video data is subject to storage, predictive analysis for simulation and recognition locally based on ubiquitous computing and embedded simulation techniques, which supports real-time intelligent transportation control.

18.3.5.3 Intelligent Training

The intelligent space for embedded computer, information equipment, and multi-mode sensor is constructed to support the intelligent training scene for data precipitation and reuse, the ubiquitous computing and AI technology.

18.4 Data-Driven M&S

Lin Zhang, Xu Xie, Longfei Zhou

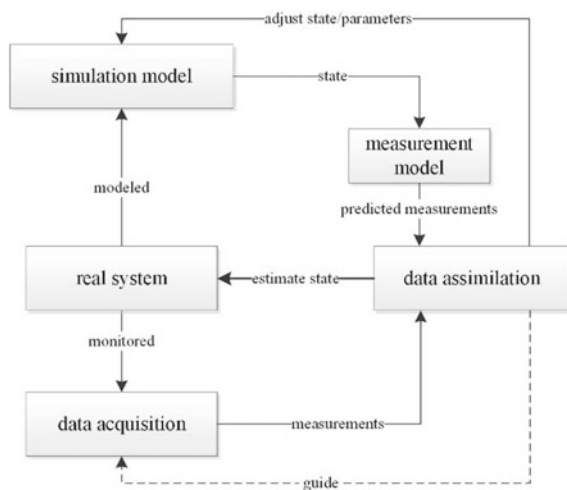
18.4.1 Introduction

Modeling and simulation are a method of choice for studying and predicting dynamic behavior of complex systems. In the application of simulation, the real system is first abstracted by a conceptual model, which is then translated into a (computer-executable) simulation model [45]. The simulation model needs to be verified (whether the computer implementation of the conceptual model is correct) and validated (whether the conceptual model can replace the real system for the purposes of experimentation) iteratively until the simulation model can represent the real system accurately, in the sense that the discrepancy between the simulation output and the relevant system measurements is within pre-specified acceptance criteria [45, 46]. During this iterative model construction process, data is used offline, e.g., for model calibration (adjusting model parameters) [47, 48], or for automated model generation [49]. Once the simulation model is verified and validated, we can experiment with the simulation model to predict the behavior of the real system, while the data itself is not used in the simulation process. However, models inevitably contain errors, which arise from many sources in the modeling process, such as inadequate sampling of the real system when constructing the behavior database for the source system [50], or conceptual abstraction in the modeling process [51]. Due to these inevitable errors, even elaborate complex models of systems cannot model the reality perfectly, and consequently, results produced by these imperfect simulation models will diverge from or fail to predict the real behavior of those systems [52, 53].

With the advancement of measurement infrastructures, such as sensors, data storage technologies, and remote data access, the availability of data, whether real-time online or archival, has greatly increased [52, 53]. This allows for a new paradigm – dynamic data-driven simulations, in which the simulation is continuously influenced by fresh data sampled from the real system [54] shows a general

dynamic data-driven simulation, which consists of (1) a simulation model, describing the dynamic behavior of the real system; (2) a data acquisition component, which essentially consists of sensors that collect data from the real system; and (3) a data assimilation component, which carries out state estimations based on information from both measurements and the simulation. The dynamic data-driven simulation integrates computational (i.e., behavior predicted by the simulation model) and measurement (i.e., real-time data from the real system collected by sensors) aspects of a system. This can lead to more accurate simulation results (i.e., the estimated model state is closer to the real system state) than using a single source of information from either the simulation model or the measurements. Integrating data from the real system also helps the simulation to prune unrealistic states, since actual system data naturally contains correlation information which is easily lost in the modeling process (e.g., by falsely assuming that state variables are independently distributed). Such an integration is achieved by an analysis technique, data assimilation, which incorporates measured observations into a dynamical system model in order to produce a time sequence of estimated system states [55, 56]. By assimilating actual data, the simulation can dynamically update its current state to be closer to the real system state, which facilitates real-time applications of simulation models, such as real-time control and analysis, real-time decision making, and understanding the current state of the real system. Besides, if the model state is extended to include model parameters, online model parameter calibration can be achieved together with the state estimation [57]. With more accurate model state and model parameters adjusted by assimilating real-time data, we can experiment (offline) on the simulation model with the adjusted state and parameters, which will lead to more accurate results for follow-on simulations. In reverse, the information from data assimilation can also be fed back to the data acquisition component to guide the measurement process, for example, to optimize the sensor deployment (Fig. 18.2).

Fig. 18.2 A general dynamic data-driven simulation



18.4.2 Data Assimilation Techniques

The aim of data assimilation is to incorporate measured (noisy) observations into a dynamic system model in order to produce accurate estimates of all the current (and future) state variables of the system [56]. Therefore, data assimilation relies on the following three elements to work, which include the system model that describes the evolution of the state over time, the measurement model that relates noisy observations to the state, and the data assimilation techniques that carry out state estimation based on information from both the model and the measurements, and in the process address measurement and modeling errors [55]. In literature, many data assimilation techniques exist, such as Kalman filter [58], extended Kalman filter [59], and ensemble Kalman filter [60]. However, their working relies on certain assumptions, such as linear model assumption, or Gaussian error assumption [61]. Another powerful data assimilation technique is the particle filters [58, 62]. The particle filters approximate a probability density function by a set of particles and their associated importance weights, and therefore, they put no assumption on the properties of the system model. As a result, they can effectively deal with nonlinear and/or non-Gaussian applications [63, 64]. The main data assimilation techniques are compared in Table 18.1.

The dynamic data-driven simulation paradigm has been successfully applied in many fields. One of the earliest applications is data assimilation in wildfire spread simulations conducted at the Systems Integrated Modeling and Simulation (SIMS) Lab at Georgia State University [64]. In this work, the simulation model for wildfire spread is a cellular automaton-based discrete event simulation model called DEVS-FIRE [65]; the measurements are temperature values from sensors deployed in the fire field; particle filters are employed to assimilate these measurements into the DEVS-FIRE model to estimate the spread of the fire front. Experimental results show that the data assimilation system is able to improve the accuracy of wildfire simulation by assimilating real-time data from sensors. The proposed framework is later applied to agent-based simulation of smart environments in order to estimate people's location information [66]. This information can help to make decisions in situations like emergency evacuation in smart environments. Other applications can be found in transportation systems in which vehicle trajectories are reconstructed by assimilating noisy event-based data and travel times into microscopic

Table 18.1 Comparison of main data assimilation techniques

	Kalman filter (KF)	Extended Kalman filter (EKF)	Ensemble Kalman filter (EnKF)	Particle filter (PF)
System model	Gaussian errors linear	Gaussian errors continuously differentiable	Gaussian errors	No restrictions
Measurement model	Gaussian errors linear	Gaussian errors continuously differentiable	Gaussian errors	No restrictions

traffic simulation models [67]. Manufacturing industrial is also an important application field of data-driven simulation. An example is presented in the next section.

18.4.3 An Example: Real-Time Scheduling of Cloud Manufacturing Services Based on Dynamic Data-Driven Simulation

Cloud manufacturing (CMfg) is a service-oriented and network-based manufacturing mode that provides customers with on-demand manufacturing services [68, 69]. The scheduling problem is critical for realizing the optimal matching and scheduling of tasks and services in CMfg [70]. Compared with typical manufacturing systems, the CMfg system has some characteristics, such as individualized requirements of customers, distributed services, and uncertainties [71]. These features bring more difficulties to CMfg than traditional environments in solving the scheduling problem.

Dynamic cloud manufacturing scheduling (DCMS) problem is different from traditional job shop scheduling problem in scopes, models, and objectives. A scheduling method based on dynamic data-driven simulation DDDS is proposed to address the DCMS problem and improve the system robustness [72].

18.4.3.1 Model of DCMS Problem

The CMfg environment differs from typical manufacturing systems in some aspects, such as tasks, services, and uncertainties. There are N service providers P_1, P_2, \dots, P_N . The logistics time between P_i and P_j is $l_{i,j}$. P_i provides n_i manufacturing services $S_{i,1}, S_{i,2}, \dots, S_{i,m_i}$. The service type of $S_{i,j}$ is $h_{i,j}$. In the dynamic CMfg environment, tasks randomly arrive at the CMfg platform. Assume that the number of arrived tasks during $[t_1, t_2]$ is $M(t_1, t_2)$, and the i th arrived task is T_i . The task type, arrival time, due date, and priority of T_i are $b_i, a_i, c_i,$ and p_i , respectively. There are a total of H different task types. The number of tasks and the service status in the CMfg system change over time. The service status includes service availability information and subtask queue information. A task can be decomposed into a subtask sequence based on its task type. The j th subtask of T_i is $I_{i,j}$ and the subtask type of $I_{i,j}$ is $g_{i,j}$. The length of the subtask sequence of task type b_i is m_i .

To study the system operation performance and the execution of tasks of different scheduling strategies, we consider the DCMS problem during a specific period of time $[t_1, t_2]$. Assume that the service time of subtask $I_{i,j}$ on $S_{x,y}$ is $e_{x,y,i,j}$. We define a relationship indicator $v_{h_{x,y},g_{i,j}}$ by (18.1) to represent the many-to-many mapping relationship from service types $h_{x,y}$ to subtask types $g_{i,j}$ as follows:

$$v_{h_{x,y},g_{i,j}} = \begin{cases} 1, & e_{x,y,i,j} < \infty \\ 0, & e_{x,y,i,j} = \infty \end{cases} \quad (18.1)$$

If service $S_{x,y}$ has the ability to execute subtask $I_{i,j}$, then relational variable $v = 1$ where $h_{x,y}$ is the type of service $S_{x,y}$ and $g_{i,j}$ is the type of subtask $I_{i,j}$. Otherwise, $v = 0$. There are three decision variables including serial number of the selected provider by $I_{i,j}$ (represented by $x_{i,j}$), serial number of the selected service by $I_{i,j}$ (represented by $y_{i,j}$) and start time of $I_{i,j}$ (represented by $z_{i,j}$).

There are multiple stakeholders in CMfg including demanders, providers, and operators. For different stakeholders, the optimization objectives of the DCMS problem may be different or even contradictory. The simulation method can emulate the operational processes of different scheduling strategies for different optimization objectives. The statistical results of simulations can assist the CMfg platform in selecting a better scheduling strategy for a specific type of stakeholder under the current system environment. We assume that the completion time of $I_{i,j}$ is $f_{i,j}$. The remaining service time of $S_{i,j}$ at t is $r_{i,j}(t)$. Total service time of subtasks in the queue of $S_{i,j}$ at t is $q_{i,j}(t)$. Completion time of T_i is F_i . The utilization rate of $S_{i,j}$ during $[t_1, t_2]$ is $u_{i,j}(t_1, t_2)$, and the delay time of T_i is d_i .

18.4.3.2 The Dynamic Scheduling Method

The advantages of simulation technologies in dynamic scheduling are reflected in predicting future scheduling performance and eventualities to guide the current scheduling decision making. Traditional simulations are largely decoupled from real manufacturing systems due to the limited availability of real-time data of manufacturing systems in the past. With the recent development of sensors and IoT-based technologies, the availability and quality of real-time data have greatly increased. In DDDS, a simulation system is continually affected by the real-time data of the real system. Ehm et al. proposed a DDDS-based optimization heuristic for the dynamic shop floor scheduling problem [73]. Keller and Hu presented a data driven simulation modeling method for mobile agent-based systems [74].

To solve the DCMS problem, a DDDS-based dynamic CMfg service scheduling method (DSS) is proposed to address the randomly arrived tasks and unpredicted service time in the dynamic CMfg environment. In this section, the DSS method is introduced from the aspects of system framework, DDDS strategy, and scheduling rules.

18.4.3.3 System Framework

The system framework of DSS is presented in Fig. 18.3. There are three main roles in the framework: service demanders, service providers, and simulation platform. Service demanders submit task requirements to the simulation platform, and simulation platform applies a service scheduling system to generate task schedules based on real-time task information, real-time service information, scheduling rules, and optimization objectives. Service providers receive allocated subtasks and execute these subtasks at specific time according to task schedules. The completed products/parts are then delivered from selected service providers to service demanders through logistics.

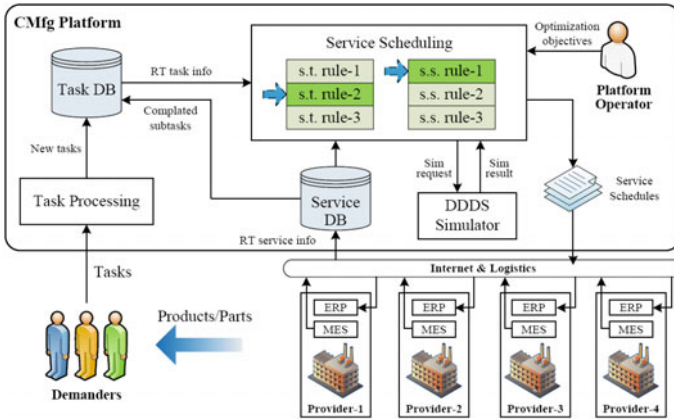


Fig. 18.3 System framework of the DSS method

18.4.3.4 Scheduling Rules

There are two kinds of scheduling rules in the proposed method including subtask priority rules and service selection rules. In the subtask priority rules, task arrival time, task due date, and subtask serial number are considered to rank the simultaneously triggered subtasks. When several subtasks compete for a single service, the task arrival times of these triggered subtasks are compared. And the scheduling program selects subtasks with earliest task arrival time. If the task arrival times of these subtasks are equal, the scheduling program compares the due dates and select subtasks with earliest due date. If the task due dates of these subtasks are equal, the scheduling program compares the subtask serial numbers and select subtasks with minimum serial numbers. If an extreme situation occurs that the task arrival times, task due dates, and subtask serial numbers of these subtasks are equal, the program randomly selects a triggered subtask.

Function values of different scheduling rules are calculated based on service time, logistics time, and queue time of candidate services. Three single rules Minimum Service Time (MS), ML Minimum Logistics Time (ML), and Minimum Queue Time (MQ), and three combined rules Minimum Logistics and Service Time (MLS), Minimum Maximum Queue and Logistics Time (MMQL) and Minimum Service, Queue and Logistics Time (MSQL) are proposed as service selection strategies. In MS, only service time is considered as the optimization criterion, and the function value of $S_{i,j}$ is $e_{i,j,k}$ where k is the type of subtask. Figure 18.4 presents an example of these scheduling rules in DSS.

The MS rule selects the service with the minimum service time to execute the current subtask $I_{2,3}$. The ML rule selects the service with minimum logistics time from service $S_{1,2}$. The total remaining service time of service $S_{i,j}$ is considered by the MQ rule. The total remaining service time of service $S_{i,j}$ is equal to the sum of the total service time of the subtasks in the queue of $S_{i,j}$ and the remaining service time of service $S_{i,j}$ at time t_1 . The MQL rule takes both the logistics time of the candidate service and the service time of the triggered subtask into account. The

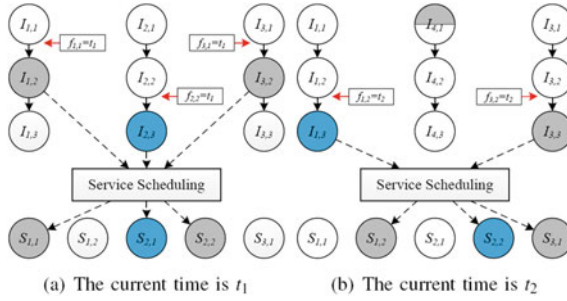


Fig. 18.4 An example of the scheduling rules in DSS method

subtask queue of service $S_{i,j}$ continues executing on $S_{i,j}$ after $I_{2,3}$ selects $S_{i,j}$ and moves by logistics. Therefore, the MMQL rule selects the candidate service with minimum max (logistics time, queue time). In MSQL, the service time, logistics time, and queue time of candidate services are considered to select the optimal service. Compared with the above five service selection rules, MSQL is of higher computational complexity.

18.4.3.5 DDDS Strategies

The interaction between the DDDS simulator and the service scheduling system in DSS is a close loop, as shown in Fig. 18.5. When trigger events occur, the service scheduling system sends the real-time service information to the DDDS simulator and makes a simulation call. The DDDS simulator runs the discrete event system specification (DEVS) simulations and then sends the evaluation results of different scheduling strategies to the service scheduling system.

The input information of the DDDS simulator includes real-time task information, real-time service information, the optimization objective and the simulation period. The real-time information of task includes the arrival time, due date, task priority, subtask numbers, completion time of completed subtasks, and the serial number of the subtask under execution. The real-time information of service includes the service busy/idle status and the remaining service time. The optimization objective is given by the platform operator to reflect the target of the current platform. The simulation period is set to be the total simulation time of the DEVS simulations.

The DEVS modeling method is applied in the DDDS simulator to model the simulation environment. The task scheduling processes of multiple distributed service providers are then simulated by the DDDS simulator. A DEVS simulation

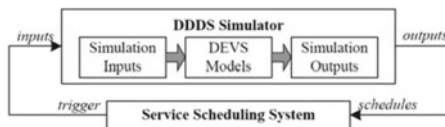


Fig. 18.5 Real-time interaction between the DDDS simulator and the service scheduling system

model includes tasks, service providers, services, and scheduling rules. The attributes of tasks include task arrival time, task types, task due date, task priority, and subtask sequence length. The attributes of service providers include service provider number N , service numbers, and logistics time. The attributes of services include service types, service capacity, and service time. There are two types of scheduling rules in the DEVS models: subtask ranking rules and service selection rules.

After being invoked, the DDDS simulator will initialize the DEVS model based on the current system status. Different scheduling rules are added to this initialized DEVS model to construct different simulation models. These simulation models with different scheduling rules are then separately simulated for the same length of time. The simulation results of different DEVS models under the current scheduling objective are then compared after these simulations. Finally, the optimal scheduling rules under the current system status are obtained and sent to the service scheduling system.

Initial results in some examples were encouraging, nevertheless further research needs to be done to confirm the feasibility and advantages of the proposed method.

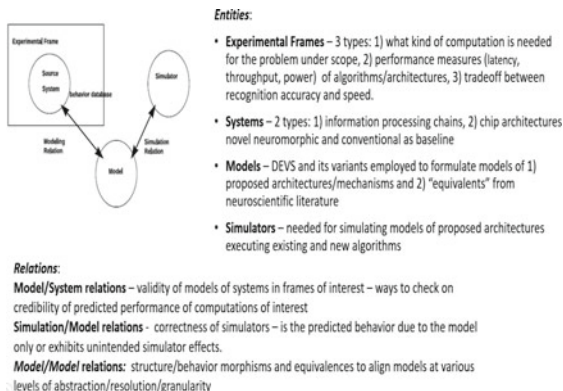
18.5 Research Enabled by the M&S Framework: Application to Neuromorphic Architecture Design

Bernard P. Zeigler

In this chapter, we illustrate how the *modeling and simulation (M&S) framework* introduced in Section 1.4.2 can help to formulate approaches to M&S in creation of novel functional products and processes. The example under focus concerns design and testing of neuromorphic architectures for a variety of information processing applications. Currently, new computer systems based on emerging understanding the brain are under serious development. Such neuromorphic architectures are intended to perform correctly in challenging environments while also meeting strict low-power consumption, time, and weight constraints. Not all modern information processing and classification algorithms can be expected to benefit significantly from migration to new form of neuromorphic hardware architectures from current computing platforms. However, the current state of the art does not readily allow prediction of neuromorphic systems' success in meeting such criteria. Indeed, an overall framework is lacking in which novel architectures can be modeled and potential applications' processing chains characterized in terms of computations required. In this chapter, we illustrate how the *modeling and simulation (M&S) framework* introduced in [12, 75] can be applied to this problem thereby illustrating how a novel domain of interest can be organized by M&S as its core basis for development.

As illustrated in Fig. 18.6, we can express an image recognition/chip machine learning (ML) [76] configuration under study within the M&S framework (Section 1.4.2) to enable its transformation into computational form. The M&S

Fig. 18.6 M&S framework in application to neuromorphic architecture design



framework allows modeling novel architectures and analyzing processing chains to be characterized in terms of computations required. This framework enables predicting the success of novel neuromorphic systems' in application to a variety of information processing applications that must meet strict low-power consumption, time, and weight constraints. For example, the framework should help to characterize a subset of modern signal processing and classification algorithms that can be expected to benefit significantly from migration from current computing platforms to new forms of neuromorphic hardware architectures. It can help to identify those types of application processing chains that are well-suited or not to such translation and the underlying reasons why this should be so.

Recall that the M&S Framework (MSF) was developed in the theory of modeling and simulation [77]. Based on systems theory, it provides a conceptual framework and an associated computational approach to methodological problems in M&S. The framework provides a set of entities and relations among the entities that, in effect, present an ontology of the M&S domain. As shown in Fig. 18.6, the entities are of four types: Experimental Frames (EF), Source Systems, Models, and Simulators, each of which is formalized as a mathematical system object that can be computationally represented in the Discrete Event System Specification (DEVS) formalism, and implemented directly in a M&S environment such as the MS4 Me Integration Platform [78] With examples given in the figure, EFs formalize the objectives of modeling studies in terms of the conditions under which models are to be simulated and the metrics of interest. Source systems are the real or to-be-designed objects for which Models are constructed to be simulated under the conditions of the EFs and thereby meet the objectives of the study. Simulators are the devices capable of executing the models, formulated as computational structures, or sets of instructions for generating observable system behaviors.

The framework's application can be analyzed in the proposed terms:

1. **Experimental Frames:** Three types of frames: (1) what kind of computation is needed for the problem under scope, (2) performance measures (latency, throughput, power) of algorithms/architectures, (3) tradeoffs between

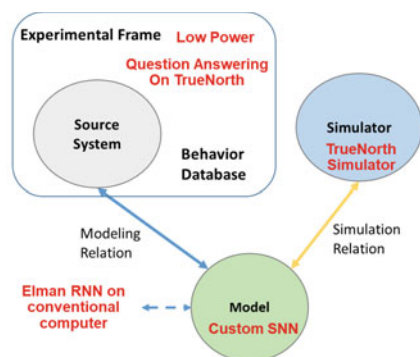
competence and performance such as between recognition/categorization accuracy and speed.

2. **Source Systems:** Existing realizations such as real-time simulation of spiking neural nets (SNNs) on SpiNNaker, Question–Answering system implemented on a TrueNorth chip, human motion and action recognition learned on Loihi chip, biomedical image learned on BrainScaleS [79].
3. **Models:** The DEVS formulations of a variety of models of (1) architectures/mechanisms, i.e., DEVS representations of neuromorphic chips, e.g., SpiNNaker, TrueNorth, Loihi, BranScaleS [80] and (2) “equivalents” from neuroscientific literature, i.e., neuroscience models such as the SNN and recurrent neural nets (RNNs) [81].
4. **Simulators:** Algorithmic characterization of devices needed for simulating models of architectures executing existing algorithms, e.g., the TrueNorth development simulator and eventual chip implementation [79], also simulation algorithms for efficient simulation of SNNs [82].

Figure 18.7 illustrates the approach and problems that arise in it. The example concerns the conversion of Elman recurrent neural networks (RNN) to spiking neural networks (SNN) for low power on IBM TrueNorth hardware [79]. An examination of the framework’s concepts suggests how this example can be analyzed in the advocated terms:

- **Source System:** Question–Answering ML on TrueNorth chip
- **Models:** Elman RNN and customized SNN derived from the original RNN.
- **Simulators:** TrueNorth development simulator and eventual chip implementation.
- **Experimental Frame:** The primary focus was on showing the lower power consumption of the converted model relative to the original implementation. The EF specification would include conditions requiring equivalent behavior in the form of question–answering categorization. In the study, an upper bound on power was derived by employing the known power consumption of the chip on which the target was implemented. This approach fits within the general concept of EF computation of performance indices.

Fig. 18.7 Framework applied to RNN-to-SNN conversion example (from literature)



In addition, the model/system, simulation/model, and model/model relations (as briefly defined in the figure (see Muzy and Zeigler [83]) are of critical importance in addressing the following:

5. **System/Model relations:** The artificial neurons of Elman RNNs were replaced by integrate-and-fire neurons of SNNs. The mapping from the RNN to the SNN was a composition of two component processes: discretization and conversion. To accommodate the network parameters on TrueNorth, the weights of the machine learning RNN had to be scaled and discretized. A rate code for the output of the recurrent layer also reduced the precision used for training. Representing the feedback structure of RNNs (that serves as a memory for non-trivial abstraction of past history) proved challenging and was approached through an approximation with error that vanishes as integration times increase. The discretization and conversion caused a drop in correct responses of some 20%. An analysis attributed portions of the drop to each of the parts of the mapping. These critical error propagation analyses can be accomplished within the concept of approximate morphism for fidelity of target model representation.
6. **Simulation/Model relations:** The model was written in MATLAB, using the integrated programming environment for IBM's Neurosynaptic System. This allows the same program to be run on a TrueNorth chip or in the associated simulation environment. This is an instance of limited model continuity [77] in that it relates only to the coded version of the source RNN. The artificial neurons of Elman RNNs were replaced by integrate-and-fire neurons of SNNs. The mapping from the RNN to the SNN was a composition of two component processes: discretization and conversion. To accommodate the network parameters on TrueNorth, the weights of the machine learning RNN had to be scaled and discretized. A rate code for the output of the recurrent layer also reduced the precision used for training. Representing the feedback structure of RNNs (that serves as a memory for non-trivial abstraction of past history) proved challenging and was approached through an approximation with error that vanishes as integration times increase. The discretization and conversion caused a drop in correct responses of some 20%. An analysis attributed portions of the drop to each of the parts of the mapping. The MSF helps to formulate these critical error propagation analyses within the concept of approximate morphism for fidelity of target model representation.

Questions

Questions arising in the design of neuromorphic architectures and the elements of the framework that are related to them are suggested in the table.

Question	Elements of DEVS NeuroMatch involved in related analysis
How general are various methods for converting existing ML neural net applications to SNN form? (In the example,	The class of models to which a method applies in relation to classes of other such methods

(continued)

(continued)

Question	Elements of DEVS NeuroMatch involved in related analysis
generality is based on agnosticism to the task on which the machine learning RNN was trained)	
What are the different structures that constrain the conversion from source to target versions of a trained model?	Model formalisms including differential equations, discrete event etc. and their characterization of computational model components: neurons, synapses, axons, etc., and behaviors: spikes, voltages, currents, etc
What are the errors involved in the current approaches to conversion from conventional to novel platforms?	Approximate morphisms that support analysis of the deviation of mappings from exact requirements of behavior preservation [84]
What accuracy reductions can be expected in current approaches to migration from conventional to novel platforms?	Error propagation analysis of approximate morphisms to predict effect of deviations from exact requirements [83]
How good is the development simulator as a predictor of performance attributes of actual hardware execution? (Typically such simulators can give only approximate statistics on the actual performance when the model is executed in the hardware itself.)	Representation of the simulator in terms of its validity for class of <i>models</i> it is aimed at
How can performance indices of the target system be estimated and related to the complexity of the task being performed?	Characterization of the structural and performance indices related to the computations required and the structure/behavior of the target system's architecture

Temporal and Dynamic Properties

Third-generation SNNs have been shown to employ temporal and dynamic properties [80] that point to new forms of applications. Introducing temporality into SNN-based computation necessitates inclusion of features that are not manifest in conventional architectures. These include timing related to the arrival of spikes, coincidence of spikes, end-to-end time of computation, and time before new inputs can be submitted/recognized by the system. A solid system-theoretical foundation and simulation framework for high-performance computational support is needed for such applications [85]. The framework helps to include system state and timing considerations:

- *Time dispersion of spikes*—how the input and output arguments are encoded in spikes over a time base, where inter-arrival times make a difference in the output [86].
- *Coincidence of spikes*—in particular, whether spikes represent arguments from the same submitted input or subsequent submission depends on their spacing in time [85].

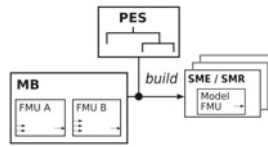


Fig.18.8 Simulator-independent FMI-based model generation in an SES/MB environment

- *End-to-end computation time*—the total processing time in a multi-component concurrent system depends on relative phasing as well as component timings, and may be poorly estimated by summing up of individual execution cycles [85].
- *Time for return to ground state*—the time that must elapse before a system that has performed a computation is ready to receive new inputs may be longer than its computation time, as it requires components to reset, i.e., return to their ground states [87].

The framework helps to formulate specific problems that arise in applications of SNNs, including:

- How to make efficient use of spike times for communication, e.g., order of arrival vs rate coding [86]?
- How to implement backpropagation through time with spiking neurons?
- How to modify traditional deep learning algorithms to use low precision numerics?
- How to model memristive device behaviors in neuromorphic computing, such as variations in conductance states with time and ambient temperature?
- How can the more advanced architectures based on an all-optical or photonic-based SNN be accommodated in the framework?

Current Neuroscience Models

The framework can apply to current neuroscience models to give potential insights into likely success of current approaches in light of emerging alternatives. Examples of such models at various system specification levels:

- **Neuron level:** Inter-spike time distribution found to be Gamma not Poisson; Spike-Timing-Dependent Plasticity (STDP) implements Hebbian learning [88].
- **Circuit Level:** Theory of Connectivity power-of-two permutation-based logic governs specific-to-general innate cell-assembly lattice arrangement (vs learned, e.g., by backpropagation) [89]
- **System level Current Neuroscience Models:** Emergence of signal propagation pathways and logic gating [90]. Bottom-up/top-down integration of information by distributed neuron microcircuits underlies high-order neural processing, e.g., perception-to-action cycle [81, 91].
- **Brain architecture:** Hierarchical multi-scale organization: neuron communities, modules, themselves are modular [92]

Conclusions

The MSF helps to develop key abstractions that help to understand the structure and behavior of various neuromorphic computation and architecture technologies within a unified conceptual modeling and simulation framework. This helps to.

- Leverage the tools of systems theory and MSF to work with both exact and approximate morphism relations between abstractions dealing with simplification, state reduction, discretization and their effects on error and its propagation.
- Take into account inherent temporal requirements of application behaviors and temporal aspects in novel learning algorithms.
- Exploit the power of the hierarchical modular DEVS formalism to (1) express the network of processing components and subcomponents across multiple sensors and (2) formulate the temporally based synchronous and asynchronous computations required to capture the relations between conventional and neuromorphic architectures.

18.6 Model Behavior Generation for Multiple Simulators

Thorsten Pawletta, Hendrik Folkerts

The research in this section builds on the concepts of the *System Entity Structure and Model Base (SES/MB) Framework* (Pawletta, Section 1.5). An aspect of this is the realization of a generic MB that allows simulator-independent model generation. This supports the exchangeability of the models. In addition, there are advantages in terms of *operational validity*. According to Sargent [93], operational validation is comprised of the simulation model and the executing simulator. Zeigler (Section 1.4.3.2) refers analogously to the *basic simulation relation* between simulator and model. Junghanns and Blochwitz [94] refer to the necessity to test simulation models with different simulators with the argument: “*DASSL of tool A differs from DASSL of tool B, even if the same name is used. Solvers contain heuristics, which are optimized with respect to the models which are most commonly simulated.*” In the field of discrete event simulation, Junglas and Pawletta [95] point out that simultaneous events in models are processed differently by simulators, resulting in different results. Comparing models with different simulators means repeating the experiment under the same conditions. The execution of experiments under equal conditions reinforces the general requirement of experiment automation. An SES/MB-based architecture for experiment automation is a further topic of this section. Here we refer to the interpretation of experiments as defined in Chapter 1 of this volume: “*Simulation is performing goal-directed experiments with models of dynamic systems.*”

According to Pawletta (Section 1.5), an SES specifies a set of system configurations with respect to the permissible system structures, associated experimental frames and possible model parameter settings. It defines the references to the dynamic basic models organized in an MB. The specification of the SES is simulator-independent. The base models are simulator-specific or generically described within model formalisms, for example, as with DEVS [77]. During the model generation, a system configuration is first derived from the SES by way of pruning and stored as *Pruned Entity Structure (PES)*. A model builder is then used to generate a simulator-dependent model based on the information from the PES and using basic models from the MB. This approach is called *native model generation* [96]. With a model formalism-based MB or different simulator-specific MBs, models can also be generated for use in different simulators. However, the effort needed to create the MBs or the model generator is relatively large.

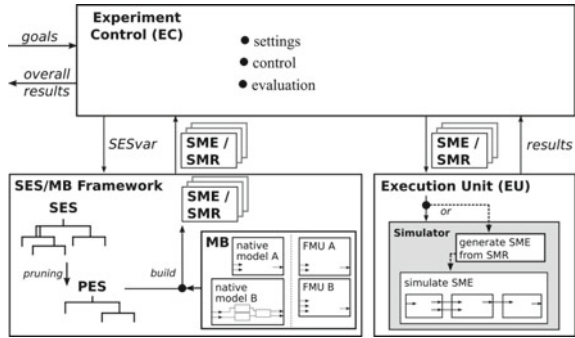
An alternative approach to building a simulator-independent MB arises from the *Functional Mock-up Interface (FMI)* standard of the automotive industry. Here, the interchangeability of the models is an urgent requirement. FMI enables the exchange of models between simulators as well as co-simulation [97, 98]. In the context of SES/MB-based M&S, we put the focus on FMI for the purpose of model exchange. A dynamic model that implements the FMI is called *Functional Mock-up Unit (FMU)*. This is especially so for the simulators in the cyber-physical system (CPS) domain that support the export and import of FMUs. Using FMI offers the possibility of specifying an easy-to-maintain MB with basic FMUs for use in multiple simulators. The simulator-independent FMI-based model generation approach is illustrated in Fig. 18.8. The MB is populated with FMUs exported from any simulator. The core of the model builder is the *build method*. The build method generates a new FMU, called *Model FMU*, according to the PES using FMUs from the MB. Using the new FMI companion standard *System Structure and Parameterization (SSP)*, the generated *Model FMU* can consist of one or more FMUs including parameterizations [99]. The FMI and SSP standards are applicable for time-continuous models, specifically signal-flow oriented and physical ones. Currently, FMI does not support models with discrete event behavior well [100].

Every M&S environment has a specific interface to import and execute models. Therefore, we differ regarding a generated simulation model between a *Simulation Model Executable (SME)* and a *Simulation Model Representation (SMR)* [101]. An SME can be executed directly by the target simulator while an SMR encodes the instructions on how to build an executable in the target simulator.

The automation of goal-directed simulation experiments requires an integration of the classic SES/MB framework into an extended architecture as shown in Fig. 18.9.

The automation requires the input and output interface of the *SES/MB Framework*. The input interface is provided by SES/MB methods such as *pruning* or *build* and additionally a set of variables with a global scope called *SESvars*. The output consists of links to the generated simulation models of type SME or SMR. For some experiments, it can be useful to derive a set of system configurations from the SES in one experiment cycle and to generate the corresponding simulation models.

Fig. 18.9 SES/MB-based architecture for automation of goal-directed experiments, derived from the basic architecture by Schmidt [102]



Automated pruning requires that the SES encodes all information to be able to derive the unique system configurations. Therefore, it is necessary to define appropriate selection rules at all variation points of the SES (Pawletta, Section 1.5). Some types of variability, such as variable coupling relationships, can easily be described in a procedural way which reduces the complexity of the tree. For this purpose, the concept of SES functions was introduced [103]. SES functions can be used for the variable parameterization of all tree attributes. For the exchange with other architectures, an XML structure was defined for the SES.

As shown in Fig. 18.9, an automated experiment is supervised by the component *Experiment Control (EC)*. The EC defines an experiment specification that can depend on the current experiment results. An experiment cycle starts with assigning values to the SESvars and using the pruning method of the SES/MB Framework. Via the SESvars, the EC initiates the derivation of a system configuration or a set of system configurations. Then the SES/MB Framework generates the respective simulation models as discussed above. The generated simulation models of type SME or SMR are either simulator-specific or simulator-independent, depending on the model generation approach used. In the next step, the EC determines for each model the execution data and methods to use such as the solver method or simulation start and stop time. This information is added to each model. The EC then initiates the execution of each model under the control of an *Execution Unit (EU)*. The EU is a kind of wrapper for one or several target simulators. Finally, the EU returns the simulation results to the EC where they are evaluated. Based on the results of the evaluation, the EC reactively decides whether a further experiment cycle is started or whether the experiment is terminated. In the latter case, the EC calculates the overall results.

An implementation of the extended SES/MB-based architecture as a Python toolset is provided on the GitHub platform [104]. A case study showing the simulator-independent FMI-based model generation is provided by Folkerts et al. [101].

18.7 Simulation-Based Disciplines

Saurabh Mittal

We live in the information age in which information underpins every technology in today's world. Scientific theories—introduced in the last five decades—are realized with today's scalable computational infrastructure effectively characterized by Moore's Law. Modeling and Simulation (M&S), along with Big Data technologies, is at the forefront of such exploration and investigation. Furthermore, simulation has a unique characteristic: Dynamic models associated with diverse experimental conditions and scenarios (sometimes in extreme and even at adverse conditions) have the power of generating new knowledge.

M&S is often taken as a single activity but in its practice and engineering, modeling and simulation are separate activities. While modeling has been adequately embraced by various disciplines, leading to many modeling techniques, many times the modeling activity is not supported by simulation activity. On the other hand, a simulation-based approach subsumes modeling as an inherent activity. While modeling helps in bringing a common understanding across all stakeholders (e.g., scientists, engineers, practitioners), it is usually through simulation that a model's correctness is evaluated. A validated model must be amenable to simulation. A model represents a real-world phenomenon using abstractions. A properly validated, verified and accurately computerized model lends itself to experimentation. Experimentation with a dynamic model as well as gaining experience based on a dynamic model lies within the domain of simulation-based approaches.

Many professionals who use simulation techniques in their practice may not consider themselves simulationists, since they primarily identify as engineers, scientists, social scientists, or medical or defense professionals. However, the technology they are applying is simulation. Simulation establishes a model in a computational environment and allows us to experiment with the model to solidify our understanding in a dynamic environment. "Computation" in computational modeling and simulation is an aspect of modeling and simulation and does not (cannot) imply that modeling and simulation is a subfield of computation or software engineering. Pretending otherwise would be like claiming that "computational astronomy" and "computational archeology," respectively, are the subfields of computation or software engineering. Rather, simulation extends the power of computation by allowing experimentation and experience. Furthermore, simulation models can act as generators of new data under the associated scenarios.

A recent book by Mittal et al. [105] contains several examples from diverse disciplines demonstrating simulation maturity to provide a solid basis for advancement in many disciplines; from life sciences to engineering; from architecture to arts to social sciences. As a sign of the times, we are truly at the crossroads where M&S itself is becoming a "discipline." The book highlights the state of the art in simulation in the modern era and how simulation-based T&E methodologies across multiple disciplines advance the very discipline itself. The notable contribution of this book is providing a comprehensive collection of

chapters from diverse disciplines with the unique characteristics of simulation. It emphasizes the fact that simulation may enhance the power of many disciplines and may serve as a foundation to shape emerging disciplines. Not only is M&S benefiting disciplines like sociology, which has largely been insulated from such experimentation, but it is also undoubtedly used in every aspect of life, be they transportation, finance, economics, biology, and so forth. Furthermore, the book explores and elaborates on the position of simulation in various disciplines and notes its impact on the advancements of these disciplines as we advance toward a computational future in the twenty-first century: a computational future to be enhanced and empowered by simulation-based approaches.

Disciplines emerge through an evolutionary process spanning a few years or decades. The first step is the generation of new knowledge from one of the following methods [106]:

1. Two or more branches of knowledge merge to form a new discipline with its own characteristics, for example, biochemistry, geobiology, neuroscience, etc.
2. A number of disciplines converge into an important field of activity and resulting in a two-way flow of ideas between pairs of disciplines that enrich both, leading to interdisciplinary knowledge, for example, transportation engineering, biomedical engineering, cyber-physical systems, etc.
3. A social or professional activity becomes an area of application for several disciplines and recognized as an independent field of study, for example, education, economics, social work, management, agriculture, engineering, etc. This leads to a complete new set of a knowledgebase completely unrelated to the constituent disciplines.
4. The changes in the sociopolitical scenario, for example, culture, urban development, smart cities, etc.
5. New research through natural discovery, subsequent development, and targeted inventions lead to a new discipline, for example, nanotechnology, space science, quantum mechanics, etc.

These methods lead to the development and massaging of the knowledgebase, which in turns defines boundaries for this specialized knowledge and the underlying taxonomy. The emerging discipline can also be categorized as multi-disciplinary, interdisciplinary or transdisciplinary. In the list above, #1 and #4 are multi-disciplinary, #2 is interdisciplinary, and #3 is transdisciplinary. The item #5 is unique and truly introduces a new scientific theory.

Model development is based on the knowledgebase of a particular discipline. It is against this knowledgebase the model is validated. While multi-disciplinary and interdisciplinary systems are grounded in constituent disciplines with established theories, the transdisciplinary and novel research-based disciplines lack the theoretic constructs to validate a model. It is in this latter case; the true vision of a *simulation-based discipline*, is realized. In these cases, the model itself becomes the vehicle to augment the theory for the new knowledgebase. In the former case, when sourced from multiple disciplines, the coming together of constituent disciplines in

a new discipline does not guarantee automatic alignment of scientific theories and paradigms. A deliberate effort is needed to align the theories and paradigms, and sometimes the disciplines cannot be reconciled. For example, developing an interdisciplinary model that brings together both the disciplines of quantum mechanics and Newtonian mechanics is infeasible, even though these two paradigms are well-established. In such a case no new disciplines emerge and when the need to bring together such disciplines is presented, the project team warrants subject matter experts from the constituent multiple disciplines with the multidisciplinary effort. When reconciled, this gives rise to a new community that aligns the paradigms and develops new technology, best practices, and knowledge-exchange mechanisms for the emerging interdisciplinary discipline. Model developed within such a discipline can now be validated with the new knowledgebase.

The exploration of the phrase “simulation-based” provides evidence that simulation-based thinking is prevalent in many disciplines including complex systems, engineering, medicine, natural sciences, physical sciences, social sciences, education, and training. Dedicated professionals spanning leadership, management, and engineering recognize M&S as a profession and as an integral structure to their day-to-day operations [107]. The simulation community includes many technical societies to serve their members in many disciplines. Simulation is a big business, with new technologies such as serious gaming, augmented reality and virtual worlds now becoming regular household entities. Furthermore, simulation used for entertainment purposes (e.g., simulation games) is also a big business. Without simulation at their core, such experiences would never have become possible.

The advancements in simulation of interdisciplinary and transdisciplinary disciplines such as cultures, human personality, and behavior as well as incorporation of social system and ethics in simulation studies may support simulation-based rational decision-making education and training as a part of the education and training of future statesmen. This may even include education of citizens including the recognition of cognitive biases leveraged by some politicians to distort reality for their personal advantages.

M&S practitioners are truly happy to see simulation now in every aspect of modern life. Today, we should not embark on any complex system study (design, analysis, or control) without considering “simulation-based” techniques unless we prefer to ignore all the benefits of simulation-based approaches. By emphasizing the role of simulation in the education of many disciplines, future professionals may be better equipped for their professions. Involving—even non-computational—simulation in K-through-12 education, future generations may be better prepared with enhanced thinking abilities.

References

1. Fishwick P (1995) Simulation model design and execution. Prentice Hall, Building Digital Worlds
2. Gentner D, Stevens AL (2014) Mental models. Psychology Press

3. Johnson-Laird P (1986) *Mental models* 1986. Harvard University Press.
4. Enderton H (2001) *A mathematical introduction to logic*. Academic Press
5. Knight K (2012) *Mind mapping: improve memory, concentration, communication, organization, creativity, and time management*. MindLily Publishing
6. Novak JD (2010) *Learning, creating and using knowledge: concept maps as facilitative tools in schools and corporations*. Routledge
7. Tolk A, Turnitsa C, Diallo S (2008) Implied ontological representation within the levels of conceptual interoperability model. *Intell Decis Technol* 2(1):3–19
8. Matthews RB, Gilbert NG, Roach A, Polhill JG, Gotts NM (2007) Agent-based land-use models: a review of applications. *Landscape Ecol* 22(10):1447–1459
9. Tolk A, Diallo SY, Padilla JJ, Herencia-Zapana H (2013) Reference modelling in support of M&S—foundations and applications. *J Simul* 7(2):69–82
10. Diallo SY, Wildman WJ, Shults FL, Tolk A (eds) (2019) *Human simulation: perspectives, insights, and applications* (vol 7). Springer, Cham
11. Robinson S, Arbez G, Birta LG, Tolk A, Wagner G (2015). *Conceptual modeling: definition, purpose and benefits*. *Proceedings of the Winter Simulation Conference, IEEE: Piscataway, NJ*. pp. 2812–2826
12. Robinson S, Nance RE, Paul RJ, Pidd M, Taylor SJ (2004) Simulation model reuse: definitions, benefits and obstacles. *Simul Model Pract Theory* 12(7–8):479–494
13. MIL-HDBK 1211 (1995) *Missile flight simulation part one surface-to-air missiles*. U.S. Department of Defense
14. STANAG. 4355 (2003) *The modified point mass and five degrees of freedom trajectory models*, Draft Edition 5.0A
15. Pidd M (2002) Simulation software and model reuse: a polemic. In: *Proceedings of the winter simulation conference, vol 1*. IEEE, pp 772–775
16. Balci O, Arthur JD, Nance RE (2008) Accomplishing reuse with a simulation conceptual model. In: *2008 winter simulation conference*. IEEE, pp 959–965
17. Reese R, Wyatt DL (1987) Software reuse and simulation. In: *Proceedings of the 19th conference on winter simulation*, pp 185–192
18. Zeigler BP, Hall SB, Sarjoughian HS (1999) Exploiting HLA and DEVS to promote interoperability and reuse in lockheed’s corporate environment. *SIMULATION* 73(5):288–295
19. Hu Y, Xiao J, Zhao H, Rong G (2013) Devsmo: an ontology of devs model representation for model reuse. In: *Proceedings of the 2013 winter simulation conference: simulation: making decisions in a complex world*, pp 4002–4003
20. Garro A, Falcone A (2015) On the integration of HLA and FMI for supporting interoperability and reusability in distributed simulation. In: *Proceedings of the symposium on theory of modeling & simulation: DEVS integrative M&S symposium*, pp 9–16
21. Exel L, Frey G, Wolf G, Oppelt M (2014) Re-use of existing simulation models for DCS engineering via the Functional Mock-up Interface. In: *Proceedings of the 2014 IEEE emerging technology and factory automation (ETFA)*. IEEE, pp 1–4
22. Bell D, de Cesare S, Lycett M, Mustafee N, Taylor SJ (2007) Semantic web service architecture for simulation model reuse. In: *11th IEEE international symposium on distributed simulation and real-time applications (DS-RT’07)*. IEEE, pp 129–136
23. Bocciarelli P, D’Ambrogio A, Giglio A, Paglia E (2019) A microservice-based approach for fine-grained simulation in msaas platforms. In: *Proceedings of the 2019 summer simulation conference*, pp 1–12.
24. Szabo C, Teo YM (2007) On syntactic composability and model reuse. In: *First Asia international conference on modelling & simulation (AMS’07)*. IEEE, pp 230–237
25. Diallo SY, Herencia-Zapana H, Padilla JJ, Tolk A (2011) Understanding interoperability. In: *Proceedings of the 2011 emerging M&S applications in industry and academia symposium*, pp 84–91

26. Bell D, Mustafee N, de Cesare S, Taylor SJ, Lycett M, Fishwick PA (2008) Ontology engineering for simulation component reuse. *Int J Enterp Inf Syst (IJEIS)* 4(4):47–61
27. Durak U, Oğuztüzün H, Köksal Algin C, Özdikiş Ö (2011) Towards interoperable and composable trajectory simulations: an ontology-based approach. *J Simul* 5(3):217–229
28. Tolk A, Diallo S, Turnitsa C (2007) Applying the levels of conceptual interoperability model in support of integratability, interoperability, and composability for system-of-systems engineering. *J Syst Cybern Inform* 5(5):65–74
29. Handler (2019) Automatic innovation: ubiquitous simulation grid technology. Baidu
30. Hill R, Al-Muhtadi J, Campbell R, Kapadia A, Ranganathan A (2004) A middleware architecture for securing ubiquitous computing cyber infrastructures. *IEEE Distrib Syst Online* 5(9):1–1
31. Li BH, Chai XD, Zhang L et al (2018) Preliminary study on modeling and simulation technologies for new artificial intelligent systems. *J Syst Simul* 30(2):349–362
32. Li BH (2005) Some focusing points in development of modern modeling and simulation technology. *Lecture Notes Comput Sci* 3398(9):12–22
33. Li N, Xu LJ, Peng XY et al (2008) Study on ubiquitous simulation system architecture and key technologies. *J Syst Simul* 16:131–135
34. Tang Z, Li BH, Chai XD (2008) Application of context-awareness in pervasive simulation grid. *Comput Integr Manuf Syst* 08:96–104
35. Tang Z, Li BH, Chai XD et al (2008) Study on ubiquitous simulation grid. In: *Computer integrated manufacturing systems*. 08
36. Tang Z (2007) Study on ubiquitous simulation grid and key technologies. Beihang University
37. Zhai Y, Sun W, Bao T, Yang K, Qing D (2018) Edge-side simulation method and framework based on micro-services. *J Syst Simul* 30(12):44–53
38. Sandhu R, Thomas RK (2004) Models, protocols, and architectures for secure pervasive computing: challenges and research directions. In: *Proceedings of the second IEEE annual conference on pervasive computing and communications workshops*. IEEE
39. Xu WS, Xin YW, Lu GZ (2007) Research and development of pervasive computing middleware technology. *Comput Sci* 34(11):1–5
40. Wu Q (2006) Research on model and methodology of adaptive middleware for ubiquitous computing. Zhejiang University
41. Tang Z, Li BH, Chai XD et al (2009) Studies on simulation service migration technologies in pervasive simulation grid. *J Syst Simul* 12:3631–3640
42. Maffioletti S, Kouadri MS, Hirsbrunner B (2004) Automatic resource and service management for ubiquitous computing environments. In: *IEEE conference on pervasive computing & communications workshops*. IEEE
43. Yau SS, Zhang X (2005) A middleware service for secure group communication in mobile ad hoc networks, vol 76. Elsevier Inc., pp 29–43
44. Zheng, D (2009) Research on key technologies of component oriented middleware based on the context-aware service. National University of Defense Technology
45. Banks J (1998) *Handbook of simulation—Principles, methodology, advances, applications, and practice*. Wiley
46. Sargent R.G. (2011). *Verification and Validation of Simulation Models*. Proc.of the 2011 Winter Simulation Conference. (S. Jain, R.R. Creasey, J. Himmelspach, K.P. White, and M. Fu, eds.), pp. 183–198.
47. Ciuffo B, Punzo V, Montanino M (2012) The calibration of traffic simulation models. Report on the assessment of different goodness of fit measures and optimization algorithms. MULTITUDE Project-COST Action TU0903. Technical report, European Commission-Joint Research Centre
48. Kesting A, Treiber M (2008) Calibrating car-following models by using trajectory data: methodological study. *Transp Res Rec: J Transp Res Board* 2088:148–156
49. Huang Y (2013) Automated simulation model generation. Ph.D. thesis, Delft University of Technology

50. Zeigler BP, Praehofer H, Kim TG (2000) *Theory of modeling and simulation: integrating discrete event and continuous complex dynamic systems*, 2nd edn. Academic Press
51. Lahoz WA, Khattatov B, Menard R (2010) *Data assimilation: making sense of observations*, 1st edn. Springer, Berlin, Heidelberg
52. Darema F (2004) Dynamic data driven applications systems: A new paradigm for application simulations and measurements. In: Bubak M, van Albada GD, Sloot PMA, Dongarra J (eds) *Computational science—ICCS 2004*. Springer, Berlin, Heidelberg, pp 662–669
53. Darema F (2005) Dynamic data driven applications systems: New capabilities for application simulations and measurements. In: Sunderam VS, van Albada GD, Sloot PMA, Dongarra JJ (eds) *Computational Science—ICCS 2005*. Springer, Berlin, Heidelberg, pp 610–615
54. Hu X (2011) Dynamic data driven simulation. *SCS M&S Mag* II(1):16–22
55. Bouttier F, Courtier P (1999) *Data assimilation concepts and methods*. Meteorological Training Course Lecture Series, ECMWF (European Centre for Medium-Range Weather Forecasts)
56. Nichols NK (2003) *Data assimilation: aims and basic concepts*. Springer, Netherlands, Dordrecht, pp 9–20
57. Bai F, Guo S, Hu X (2011) Towards parameter estimation in wildfire spread simulation based on sequential Monte Carlo methods. In: *Proceedings of the 44th annual simulation symposium*, Boston, MA, USA, pp 159–166
58. Arulampalam MS, Maskell S, Gordon N, Clapp T (2002) A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Trans Signal Process* 50(2):174–188
59. Gillijns S, Mendoza O, Chandrasekar J, De Moor BLR, Bernstein D, Ridley A (2006) What is the ensemble Kalman filter and how well does it work? In: *Proceedings of the 2006 American control conference*, Minneapolis, MN, USA, pp 4448–4453
60. Evensen G (2003) The ensemble Kalman filter: theoretical formulation and practical implementation. *Ocean Dyn* 53(4):343–367
61. Yuan Y (2013) *Lagrangian multi-class traffic state estimation*. Ph.D. thesis, Delft University of Technology
62. Djurić, PM, Kotecha JH, Zhang J, Huang Y, Ghirmai T, Bugallo MF, Miguez J (2003) Particle filtering. *IEEE Signal Process Mag* 20(5):19–38
63. Gu F, Hu X (2008) Towards applications of particle filters in wildfire spread simulation. In: *Proceedings of the 2008 winter simulation conference*, Miami, FL, USA, pp 2852–2860
64. Xue H, Gu F, Hu X (2012) Data assimilation using sequential Monte Carlo methods in wildfire spread simulation. *ACM Trans Model Comput Simul* 22(4): 1–23, 25
65. Hu X, Sun Y, Ntaimo L (2012) DEVS-FIRE: design and application of formal discrete event wildfire spread and suppression models. *SIMULATION: Trans Soc Model Simul Int* 88(3):259–279
66. Wang M, Hu X (2015) Data assimilation in agent based simulation of smart environments using particle filters. *Simul Model Pract Theory* 56:36–54
67. Xie X, van Lint H, Verbraeck A (2018) A generic data assimilation framework for vehicle trajectory reconstruction on signalized urban arterials using particle filters. *Transp Res Part C: Emerg Technol* 92:364–391
68. Li BH, Zhang L, Chai XD (2010) *Introduction to cloud manufacturing*, no. 4. ZTE Communications
69. Zhang L, Luo YL, Tao F, Li BH, Ren L, Zhang XS, Guo H, Cheng Y, Hu AR (2014) Cloud manufacturing: a new manufacturing paradigm. *Enterp Inf Syst* 8(2):167–187
70. Li F, Liao TW, Zhang L (2019) Two-level multi-task scheduling in a cloud manufacturing environment. *Robot Comput Integr Manuf* 56:127–139
71. Li F, Zhang L, Liao TW, Liu YL (2019) Multi-objective optimisation of multi-task scheduling in cloud manufacturing. *Int J Prod Res* 57(12):3847–3863

72. Zhou LF, Zhang L, Ren L, Wang J (2019) Real-time scheduling of cloud manufacturing services based on dynamic data-driven simulation. *IEEE Trans Ind Inf* 15(9)
73. Kück M, Ehm J, Hildebrandt T, Freitag M, Frazzon EM (2016) Potential of data-driven simulation-based optimization for adaptive scheduling and control of dynamic manufacturing systems. In: 2016 winter simulation conference, Washington, DC, pp 2820–2831
74. Keller N, Hu X (2016) Data driven simulation modeling for mobile agent-based systems. In: 2016 symposium on theory of modeling and simulation, Pasadena, CA, pp 1–8
75. Zeigler BP, Muzy A, Kofman E (2019) Introduction to systems modeling concepts. Theory of modeling and simulation. Academic Press, Orlando, pp 3–25
76. Chen S, Wang H (2014) SAR target recognition based on deep learning. In: Proceedings of the 2014 IEEE international conference on data science and advanced analytics, pp 541–547
77. Zeigler BP, Muzy A, Kofman E (2018) Theory of modeling and simulation: discrete event & iterative system computational foundations, 3rd edn. Elsevier
78. Seo C, Zeigler BP, Coop R, Kim D (2013) DEVS modeling and simulation methodology with MS4 Me software. In: Symposium on theory of modeling & simulation, Spring Sim San Diego
79. Diehl PU, Zarrella G, Cassidy A (2016) Conversion of artificial recurrent neural networks to spiking neural networks for low-power neuromorphic hardware. [arXiv:1601.04187](https://arxiv.org/abs/1601.04187) [cs:NE]
80. Rajendran B, Sebastian A (2019) Low-power neuromorphic hardware for signal processing applications: a review of architectural and system-level design approaches. *IEEE Signal Process Mag* 36(6):97–110. <https://doi.org/10.1109/MSP.2019.2933719>
81. Opris I, Casanova MF (2017) Prefrontal cortical microcircuits support the emergence of mind. In: Springer series in cognitive and neural systems, vol 11
82. Grinblat GL, Herman A, Kofman E (2011) Quantized state simulation of spiking neural networks. *Simulation: Trans Soc Model Simul Int* 88(3):299–313
83. Muzy A, Zeigler BP (2020) Morphisms for lumping finite-size linear system realizations of componentized neural networks, <https://hal.archives-ouvertes.fr/hal-02429240v4>
84. Jarvis D (2020) Machine learning of an approximate morphism of an electronic warfare simulation component by <https://springssim.conferencespot.org/event-data>. Accessed 20 Jan 2020
85. Zeigler BP, Muzy A (2017) Temporal modeling of neural net input/output behaviors: the case of XOR. *Systems* 5(1):7
86. Panda P, Srinivasa N (2018) Learning to recognize actions from limited training examples using a recurrent spiking neural model. *Front Neurosci* 12:126. <https://doi.org/10.3389/fnins.2017.00126>
87. Pascanu R, Mikolov T, Bengio Y (2013) On the difficulty of training recurrent neural networks. In: International conference on machine learning. jmlr.org
88. Li M et al (2018) Spike-timing pattern operates as gamma-distribution across cell types. Accessed 20 Jan 2020
89. Xie K (2016) Brain computation is organized via power-of-two-based permutation logic frontiers. *Neuroscience*. <https://doi.org/10.3389/fnsys.2016.00095>
90. Abbott TP, Vogels LF (2005) Signal propagation and logic gating in networks of integrate-and-fire neurons. *J Neurosci* 25:10786–10795
91. Zeigler BP (2020) Hybrid iterative system specification of cyberphysical systems: neurocognitive behavior application. *SpringSim*
92. Petersen SE, Sporns O (2018) Brain networks and cognitive architectures, vol 88, Issue 1, pp 207–219
93. Sargent RG (2011) Verification and validation of simulation models. In: Proceedings of the 2011 winter simulation conference, Phoenix, AZ, USA, pp 183–198
94. Junghanns A, Blochwitz T (2017) FMI is great—But not magic. In: FMI user meeting, 12th international modelica conference, 15–17 May 2017, Prague, Czech Republic. <https://fmi-standard.org/literature/>. Accessed 20 Nov 2020

95. Junglas P, Pawletta T (2019) Non-standard Queuing Policies: Definition of ARGESIM Benchmark C22. *SNE—Simulation Notes Europe* 29(3):111–115. <https://doi.org/10.11128/sne.29.bn22.10481>
96. Folkerts H, Pawletta T, Deatcu C, Hartmann S (2019) Python-based eSES/MB framework: model specification and automatic model generation for multiple simulators. *SNE Simul Notes Europe* 29(4):207–215
97. Blochwitz T, Otter M, Arnold M, Bausch C, Clauß C, Elmqvist H, Junghanns A, Mauss J, Monteiro M, Neidhold T, Neumerkel D, Olsson H, Peetz JV, Wolf S (2011) The functional mockup interface for tool independent exchange of simulation models. In: Proceedings of the 8th international modelica conference. Modelica conference, 2011, March, Dresden, Germany, pp 105–114. <https://doi.org/10.3384/ecp11063105>
98. Blochwitz T, Otter M, Akesson J, Arnold M, Clauß C, Elmqvist H, Friedrich M, Junghanns A, Mauss J, Neumerkel D, Olsson H, Viel A (2012) Functional mockup interface 2.0: the standard for tool independent exchange of simulation models. In: Proceedings of the 9th international modelica conference. Modelica conference, 2012, Sept, Munich, Germany, pp 173–184. <https://doi.org/10.3384/ecp12076173>
99. Modelica Association (2020). Modelica Association project system structure and parametrization (SSP). Modelica Association c/o PELAB, IDA, Linköpings Universitet, Linköping, Sweden
100. Cremona F, Lohstroh M, Broman D, Lee EA, Masin M, Tripakis S (2019) Hybrid co-simulation: it's about time. *SoftwSyst Model* 18:1655–1679. <https://doi.org/10.1007/s10270-017-0633-6>
101. Folkerts H, Pawletta T, Deatcu C (2021) Model generation for multiple simulators using SES/MB and FMI. *SNE - Simulation Notes Europe* 31(1):25–32. <https://doi.org/10.11128/sne.31.tn.10554>
102. Schmidt A (2018) Variantenmanagement in der Modellbildung und simulation unter Verwendung des SES/MB frameworks [Variant management in modeling and simulation using the SES/MB framework]. Ph.D. thesis, ASIM FBS—Advances in Simulation No. 30, ARGESIM Publisher Vienna, Austria, 10.111.28/fbs.30
103. Pawletta T, Schmidt A, Zeigler BP, Durak U (2016) Extended variability modeling using system entity structure ontology within MATLAB/simulink. In: Proceedings of SCS International SpringSim/ANSS 2016, Pasadena/CA, USA, SCS, pp 62–69
104. RG CEA (2020) Python-based SES/MB infrastructure. Research Group CEA, Wismar University of Applied Sciences. https://www.github.com/cea-wismar/SESMB_Inf_Python. Accessed 17 Sep 2022
105. Mittal S, Durak U, Ören T (2017) Guide to simulation-based disciplines: advancing our computational future. Springer AG
106. Areekkuzhiyil S (2017) Emergence of new disciplines. *Edutracks* 17(4):20–22
107. Tolk A, Ören T (2017) The profession of modeling and simulation: discipline, ethics, education, vocation, societies, and economics. Wiley



Trends, Desirable Features, and Challenges

19

Bo Hu Li, Wenhui Fan, and Lin Zhang 

Abstract

The SCS M&S Body of Knowledge is closed by a chapter on trends, desirable features, and challenges. Where are we going with simulation? How are other supporting disciplines evolving? What are current simulation technology trends? The following section on desirable features elaborates on the ideal characteristics set to make modeling and simulation technology rapidly develop into a generic and strategic technology. The chapter concludes with technical and conceptual challenges that will have to be addressed soon, hopefully contributing to the next iteration of the BoK.

Keywords

Modeling and simulation · Simulation technology · Digital twins · Edge simulation · Intelligent M&S · High-performance simulation · System-of-systems

B. H. Li (✉) · L. Zhang
Beihang University, Beijing, China
e-mail: bohuli@moon.bjnet.edu.cn

L. Zhang
e-mail: johnlin9999@163.com

W. Fan
Tsinghua University, Beijing, China
e-mail: fanwenhui@mail.tsinghua.edu.cn

19.1 Trends

Bo Hu Li, Wenhui Fan, Lin Zhang

A new round of revolution in technology and industry is spreading worldwide nowadays. Driven by the vision of this new era for innovative, coordinated, green, open, and shared development, it is ushering in major changes to all sectors of society in terms of the new model, means, and business format. A new landscape that is all connected, intelligence-led, data-driven, service-shared, inter-sector integrated, and massive innovative is being shaped as a result of deep integration of various technologies into every corner of the national economy, people's livelihood and national security. These include new Internet technology (Internet of things, Internet of vehicles, mobile Internet, satellite network, space-ground integrated information network, new-generation Internet, etc.), new information and communication technology (cloud computing, big data, fifth-generation mobile communication (5G), high-performance computing, modeling/simulation, digital twin, blockchain, quantum computing, etc.), new artificial intelligence technology (data-driven deep reinforcement learning intelligence, network-based swarm intelligence, technology-oriented hybrid intelligence of human-computer and brain-computer interaction, cross-media reasoning intelligence, autonomous intelligent unmanned system, etc.), new energy technology (solar energy, wind energy, bioenergy, hydrogen energy, geothermal energy, marine energy, chemical energy, nuclear energy, etc.), new material technology (metal material, inorganic non-metal material, organic polymer material, and advanced composite material technologies), and new biotechnology (new biomedicine, green bio-manufacturing technology, advanced biomedical materials, cutting-edge generic biotechnology, biological resource utilization, biosecurity, life science instruments and equipment, synthetic biology, biological big data, regenerative medicine, 3D bioprinting, etc.).

Modern modeling and simulation technology is an essential part of the new information and communication technology. Modeling and simulation technology, driven by various application requirements and related discipline development, has evolved into a new comprehensive technical system and quickly grown into a general and strategic technology. It is serving as the third vital means to understand and transform the objective world after theoretical research and experimental research. Its research falls into three major categories: simulation modeling theory, methods and technologies; simulation systems and supporting technologies; and simulation application engineering technologies (see Fig. 19.1).

At present, modeling and simulation technology is developing toward the new "digital/data-oriented, virtual/augmented reality-based, high-performance/parallel, networked/cloud-based, intelligent, and universal" trend.

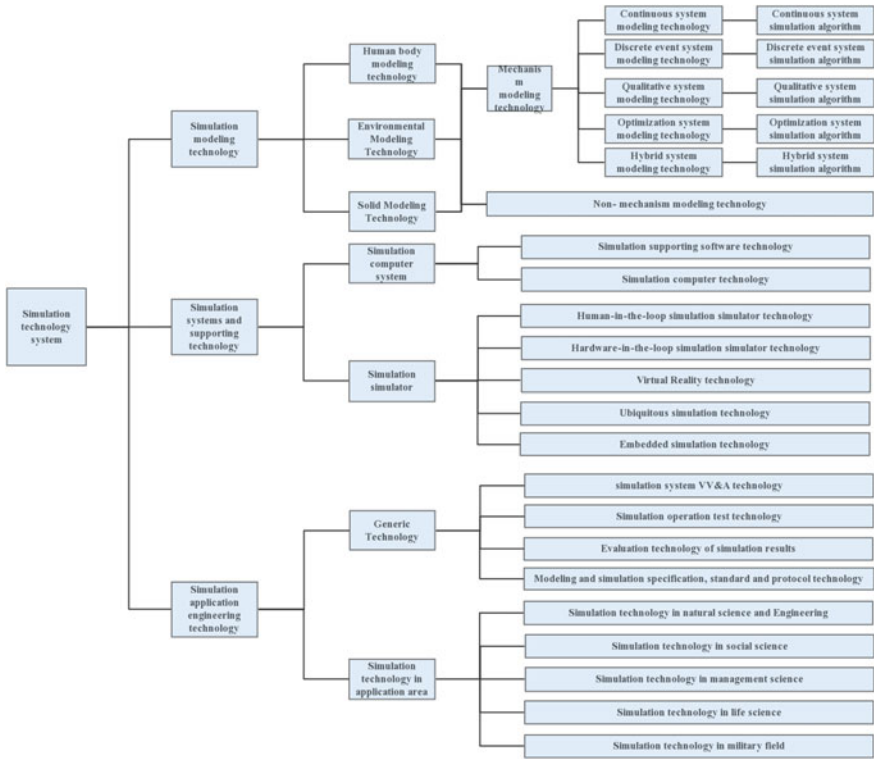


Fig. 19.1 Simulation technology system

19.1.1 New Digital/Data-Based Modeling and Simulation Technology

New digital/data-based modeling and simulation technology is a new kind of technology that integrates new digital technologies, new data processing technologies, and modeling and simulation technologies [1]. In addition to improving existing mechanism-based digital modeling and simulation technologies (such as the DEVS theory and methods in continuous development, and Petri Net modeling methods), modeling and simulation technology based on new data processing technologies will be a focus of the future development. Because modern complex large-scale systems are usually nonlinear, time-varying, multivariable, and uncertain, it is difficult to build an accurate model based on mechanisms, making it challenging to analyze and predict the behavior of complex large-scale systems realistically. Modeling and simulation technology based on new data processing technologies provide new ideas for solving such system simulation. New digital/data-based modeling and simulation technology mainly includes new-generation digital modeling, dynamic data-driven simulation technology, man-machine-material fusion simulation technology, digital twin technology, etc.

As one of the critical supporting technologies for the new-generation intelligent systems [2], the new-generation digital modeling (NGDM) [3] combines typical M&S and new information technologies, such as CPS, IoT, AI, and VR. For specific simulation needs, a highly reliable NGDM needs to be constructed based on the structure and real-time status of the physical system. NGDM can also be built based on data correlation rather than system-based mechanisms (such as artificial neural network models). With NGDM, not only offline system analysis and prediction but also online interaction with physical systems are made possible. As a typical NGDM, digital twin (DT) is an extension and development of traditional virtual prototype (VP) technology. VP is built for design. When a product is constructed, VP is turned into DT along with the data sensed from the physical product [3, 4]. Although DT cannot be the same as the physical object, it is possible to construct a suitable DT model according to different simulation requirements to adapt to different scenarios. Siemens recommend that the enterprise DT includes product DT, production process DT, and equipment DT. With these three types of DT integrated, the actual production system is simulated, tested, and optimized through a virtual enterprise based on models and automation technology before the actual production is conducted [5].

Digital twin technology is a kind of new simulation technology that makes full use of the static and dynamic operation data of the simulated physical/logical system, integrates the new digital technology, new data processing technology, and multi-disciplinary, multi-physical, multi-scale, virtual-real mapping technologies, and completes the corresponding physical system/logical system/physical equipment life cycle activities in the virtual space in a digital way. Its research scope covers perception, model construction, virtual-real model interaction, data/model/technology/business integration, cyber-physical system control, and man-machine interaction technology. It is developing into a key technology in various cyber-physical systems and intelligent systems and has drawn great attention from academics and industry [6].

Edge simulation is to configure the corresponding computing and simulation functions at the low end (device terminal) of the Internet of things as a means to improve the real-time performance of the system and ease the pressure of the cloud platform from management and control of the entire system. It is a supplement and enhancement to cloud simulation. The collaborative performance between edge simulation and cloud simulation provides more comprehensive support for production process management, scheduling, and optimization. For embedded simulation, the simulation system is embedded in the actual system and participates in the system operation. Embedded simulation technology was initially developed to meet military training requirements. Simulation modules are embedded in real combat equipment in order to support simulation and training of operators in real systems and improve their simulation experience. In the intelligent manufacturing environment, embedded simulation can support real-time scheduling, site monitoring, quality inspection, and situation prediction. In addition, embedded simulation can be used to train workers on how to use complex manufacturing equipment. Embedded simulation is an extension of edge simulation. Both edge simulation and

embedded simulation in intelligent manufacturing are relatively new research orientations. There are some key issues that need to be addressed.

19.1.2 New Virtual/Augmented Reality Modeling and Simulation Technology

New virtual/augmented reality modeling and simulation technology is a type of new technology that combines new virtual/augmented reality technology and modeling and simulation technology. It is furthered not only by the advancement of related technologies but also by the requirements from economic development (such as industrial manufacturing, cultural entertainment, education and training, and e-commerce), social development (such as public safety and urban management), national defense development (such as weapon development, combat simulation, and personnel training), and scientific development (such as natural science, social sciences, management science, life science, and engineering science).

New virtual/augmented reality modeling and simulation technology is a comprehensive simulation technology developed based on computer graphics, computer image processing, computer vision, network communication technology, computer simulation technology, multimedia technology, psychology, cybernetics, electronics, and other disciplines and technologies. It features the construction of a unified and complete virtual environment for the entire system. To represent the real characteristics of the objective world, it enables a large number of entities integrated and controlled in the virtual environment to interact with each other or with the virtual environment itself. It can simulate real entities (people, objects) and the environment in terms of vision, hearing, touch, and function, performance, and behavior. With 3I (immersion, imagination, and interaction), it satisfies the development needs and technical directions of modern simulation technology [10].

The research scope of new virtual/augmented reality modeling and simulation technology falls into five categories: perception technologies (such as vision, hearing, smell, taste, force, and touch), modeling technologies (such as geometric modeling, image modeling, physical modeling, and behavior modeling), presentation technologies (such as virtual reality rendering, naked eye 3D display, and helmet-mounted display), interaction technologies (such as human-computer interaction, environmental interaction, and virtual reality interaction), and platform-based integration technologies. In the field of manufacturing, virtual reality (VR) is mainly used for design and test, while augmented reality (AR) is usually used for assembly. In the process of assembly, the instruction information of product assembly can be displayed in the operator view by wearing AR glasses, which is very important for assisting the operator in assembling products. Park [7] proposed an AR-based model that can interactively change product shape as well as color, texture, and user interface. Ng et al. [8] proposed a gesture-based AR design environment in which designers can evaluate and modify their 3D models through gestures.

19.1.3 New High-Performance/Parallel Modeling and Simulation Technology

New high-performance/parallel modeling and simulation technology is a type of new technology that combines new high-performance, high-reliability, parallel computing technology with modeling and simulation technology. Complex system simulations such as national and national defense strategy research, emergency response processing, transportation/communication network simulation, aviation scheduling, virus transmission mechanism, weapon and equipment system demonstration, and battle plan analysis and evaluation often include a large number of entities, and there are intricate and complex interactions between entities. With the continuous development of simulation application, the simulation will be built on a larger scale and in a more complex manner, and it will demand more computing resources. On the other hand, due to the randomness of the simulation process, the simulation of complex systems often entails an exploration of the uncertainty factors within the parameter space of a large sample by traversing various parameter combinations. That requires the simulation to run hundreds of times or even more for a single analysis, evaluation, demonstration, etc. If a single simulation runs for a long time, therefore, it will take a rather long time to make a single analysis, evaluation or demonstration. This monotonous and lengthy simulation operation not only wastes valuable human and material resources but also hinders the development of complex system research and the improvement of research capabilities. For this reason, high-performance parallel simulation is becoming an essential direction for the development of this type of simulation [10].

The research scope of new high-performance/parallel modeling and simulation technology covers: (1) parallel algorithms, including how to achieve dynamic load balancing; how to reduce the amount of communication required for collaboration in parallel computing; how to take into account the scalable, portable, large-grained task-level parallel and the reasonable data structures, programming, and communication methods in each process that facilitate the performance of stand-alone machine; (2) distributed parallel algorithms, including the distributed algorithms based on grid and cloud simulation platforms, and the distributed parallel discrete system algorithms; and (3) the parallel programming environment. At present, it is urgent to study the user-oriented parallel programming environment which is easy to use, reliable, and scalable. Parallel algorithms and parallel programming are the scientific issues and critical technologies in which breakthrough must be made in the next few decades.

19.1.4 New Networked/Cloud-Based Modeling and Simulation Technology

New networked/cloud-based modeling and simulation technology is a type of new technology that combines new network technology, new cloud computing technology, and modeling and simulation technology. Modeling and simulation (M&S)

faces the following problems in various engineering and non-engineering fields: (1) the expansion and complexity of simulated systems in terms of scale and structure, and the multi-unit collaborative simulation in need of a distributed system with distributed, heterogeneous, collaborative, interoperable, and reusable functions; (2) the availability of the required simulation service to various users via the Internet. Driven by the above demands, “networked/cloud-based simulation” has come into being. Networked/cloud-based simulation is performed on the network, including three types of simulation: (1) the simulation operation of the model through the network interconnection, which is a traditional networked simulation, represented by DIS, ALSP, and HLA; (2) completion of simulation experiments through networked collaboration, represented by WEB-based simulation and HLA Evolved; and (3) the network-based domain simulation service environment, which is aimed to achieve overall and efficient simulation goals such as complex system modeling, simulation operation, and result analysis, and represented by cloud simulation and domain simulation engineering. As the real world tends to be complex nowadays, networked simulation finds its demand from the complex system problems faced by the national development [10].

The core of networked/cloud-based simulation is to build a simulation environment that supports collaboration, sharing, and service in the application field and create a win-win simulation ecosystem for stakeholders. At present, its research content can be summarized as follows: (1) a type of networked/cloud-based simulation technology with advanced simulation modes, theories and methods; (2) a type of network/cloud-based simulation technology with high efficiency and quality to complete all activities in the whole life cycle of simulation; (3) a type of networked/cloud-based simulation technology that enables a logically centralized and physically decentralized simulation environment, overcoming through the network such barriers as are caused to inter-departmental collaboration by geographical locations; (4) a type of networked/cloud-based simulation technology that emphasizes the collaboration and resource sharing services among various departments in the application field, enhances the simulation application capability and realizes low-cost and efficient design, analysis, evaluation, and production; (5) a type of networked/cloud-based simulation technology that can be used to build simulation systems with multiple forms and functions as required by different needs.

19.1.5 New Intelligent Modeling and Simulation Technology

New intelligent modeling and simulation technology is a type of new technology that combines new intelligent technology with modeling and simulation technology. New intelligent technology includes brain science, cognitive science, and artificial intelligence technology. In recent years, driven by the rapid development of the new-generation artificial intelligence technology, the development of intelligent modeling and simulation technology has reached a new height. There are two important research and application areas for new intelligent modeling and

simulation technology. The first is the modeling and simulation of intelligent systems (such as human systems) and artificial intelligence systems (such as brain-like intelligent robots and complex adaptive systems). The second is the use of artificial intelligence technology to facilitate simulation modeling, operation and result from analysis and evaluation [9].

The research hotspots of new intelligent modeling and simulation technology include: simulation modeling theory, methods, and technologies (primary modeling such as qualitative and quantitative hybrid system modeling oriented to simulated objects, metamodel framework-based modeling, variable structure modeling, big data-based modeling, deep learning-based simulation modeling, agent-oriented modeling, ontology-oriented modeling, and SDG modeling; secondary modeling such as high-efficiency multi-level parallel simulation algorithms, neural network-based optimization simulation algorithms, and machine learning-oriented simulation algorithms); simulation systems and supporting technologies (such as intelligent simulation clouds oriented to new artificial intelligence systems, multi-disciplinary virtual prototype engineering of complex products, intelligent problem-oriented simulation language for complex systems, intelligent simulation computer systems for edge computing, intelligent cross-media visualization technology, universal interconnection interface, and intelligent parts technology); and simulation application engineering technologies (such as full life cycle VV&A, full system VV&A, hierarchical VV&A, all-personnel VV&A, and comprehensive management VV&A, the management, analysis, and evaluation technology for intelligent system simulation experiment results, big data analysis, and evaluation technology, etc.).

19.1.6 New Ubiquitous Modeling and Simulation Technology

New ubiquitous modeling and simulation technology is a type of new technology that combines ubiquitous computing technology with modeling and simulation technology. Tightly integrating the simulator's hardware and software, communication hardware and software, various sensors and all of the simulated objects/world, it is a new simulation mode that combines the simulation space with the physical/logical space of everything. It is significant in that the simulation technology is introduced into real systems and seamlessly embedded into everything/the world, realizing pervasive simulation [10].

New ubiquitous modeling and simulation technology is a new field that needs to be vigorously developed. Its research scope covers: the advanced ubiquitous simulation system architecture that integrates distributed simulation technology, grid computing technology, cloud computing technology, and ubiquitous computing technology; the development of software platforms and middleware for ubiquitous simulation; establishing new interactive channels for human and simulation computing services; establishing new simulation application models for ubiquitous computing mode; providing new simulation services suitable for the needs of ubiquitous computing; the coordinated management and integration technology for

simulation space and physical/logical space of everything; self-organization, self-adaptability, and high-fault tolerance of ubiquitous simulation based on ubiquitous computing; ubiquitous simulation application technology, etc.

19.2 Desirable Features

Bo Hu Li, Wenhui Fan, Lin Zhang

The ideal characteristics are set to make the modeling and simulation technology rapidly develop into a generic and strategic technology, a third ideal means to understand and transform the objective world subsequent to theoretical research and experimental research, as a result of various demand for modeling and simulation in various applications as well as of the development of related technologies.

19.2.1 Ideal Characteristics in Simulation Modeling Theory, Method, and Technology

The first is to form a class of entirely realistic object-oriented modeling (i.e., primary modeling) theories, methods, and techniques that enable a completely realistic modeling of animate/inanimate objects in the real or imagined world based on their continuous, discrete, qualitative, decision-making, optimization, and other complex mechanisms in the full life cycle; the second is to form a complete class of algorithms for various types of simulation computers and simulators (i.e., secondary modeling), that is, various models formed by object-oriented primary modeling, and a set of simulation modeling theories, algorithms, and technologies with high speed, high reliability, high quality, high-energy saving and high availability that are intended for all kinds of simulation computers and simulators.

19.2.2 Ideal Characteristics in Simulation Systems and Supporting Technologies

It is intended to develop the modeling and simulation system and the supporting technology that can support the vision of “innovative, coordinated, green, open, and shared development,” optimize, and independently integrate “digital/data-oriented, virtual/augmented reality-based, high-performance/parallel, networked/cloud-based, intelligent, and universal modeling” on demand [10].

19.2.3 Ideal Characteristics in Simulation Application Engineering Technology

The first is to develop a class of perfect generic simulation application engineering technology, including the VV&A technology that is applied to the full life cycle, the whole system, all personnel and all aspects of management of the models of the animate/inanimate objects in the real or imagined world; the analysis, decision-making, and visual technology that is applied to highly-efficient man/machine intelligent simulation experimental results; the second is to develop a class of perfect modeling and simulation technology oriented to such fields as scientific experiments, engineering science, social sciences, life science, management science, and military science.

19.3 Challenges

Bo Hu Li, Wenhui Fan, Lin Zhang

19.3.1 Challenges to Simulation Modeling Theory and Methods

19.3.1.1 Challenges to Virtual Reality Modeling Theory and Methods

Currently, virtual reality modeling mainly focuses on fixed topological geometric modeling and dynamic physical modeling oriented to virtual environments and objects. One of critical problems in future virtual reality to be addressed is how to establish a variable topological geometric model and a more comprehensive physical model, or even an intelligent model that can self-evolve and be so “viable” as to enable the virtual reality system. It not only to give a more comprehensive and realistic visual expression but also to present the dynamic evolution of functions and environment/events as well as the intelligent behaviors of animate objects in a more realistic manner. The real world contains complex, dynamic, multi-source, massive data. Another critical problem to be addressed for future virtual reality is how to efficiently collect these data and perform automatic analysis and real-time modeling so that the virtual reality system can genuinely express the rapidly changing real world and develop “synchronously” with the latter. It is also an intelligent modeling problem [10].

19.3.1.2 Challenges to Networked Simulation Modeling Theory and Methods

Challenges to networked simulation modeling theory and methods include: (1) Studies focused on cloud simulation theory and methods as well as new patterns and formats of “Internet+” simulation based on high-performance simulation theory

and method research; studies on the theory and methods of simulation modeling automation and intelligence; studies on the visual modeling methods of complex systems in the application fields; studies on the networked simulation experiment methods based on data cultivation; and initiation of studies on the theory and methods of intelligent networked simulation to establish a domain model framework in each application field. (2) Studies focused on the theory and methods of intelligent networked simulation; studies on the integration of simulation system with other systems; and studies on networked simulation modeling methods based on big data and artificial intelligence; initiation of studies on ubiquitous simulation theory and methods to build a standardized model library for each application field based on the domain model framework. (3) Studies focused on ubiquitous simulation theory and methods, and studies on the construction method of dynamic earth simulator; development of an interoperable domain model framework system to enable models in different application domains to achieve interoperability [10].

19.3.1.3 Challenges to Intelligent Simulation Modeling Theory and Methods

The challenges to intelligent simulation modeling theory and methods include: (1) Developing modeling methods based on big data intelligence. Due to the high complexity of the mechanism of the new artificial intelligence system, it is often difficult to establish a system principle model based on the mechanism (analytic method). Therefore, the internal mechanism needs to be simulated and emulated through a large number of experiments and application data. Big data intelligence-based modeling methods are a type of method for effectively simulating intelligent systems with ambiguous mechanisms using massive observations and application data. The main research scope covers data-based reverse design, data-based neural network training and modeling, and data clustering-based modeling. (2) Developing deep learning-based simulation modeling methods. Within the environment of the new artificial intelligence system, the data to be collected and used has exploded. At the same time, the neural network based on deep learning and simulation of the human brain provides evolutionary support for the development and application of intelligent simulation modeling. (3) Developing machine learning-oriented simulation algorithms. Machine learning methods have formed a huge pedigree. How to effectively simulate and model machine learning methods in new artificial intelligence systems and put it to extensive use will be a new research orientation of significance. (4) Developing an optimization-based simulation algorithm to perform multi-sample iterative simulation calculations [10].

19.3.1.4 Challenges to High-Performance Simulation Modeling Theory and Methods

Simulation modeling is an abstract description of the simulated objects such as entities (airplanes, missiles, ships, vehicles, machinery, etc.), natural environments (terrain, atmosphere, ocean, space), and human behaviors (individuals, groups, organizations). Mechanism modeling is used in most application fields to describe the characteristics and behaviors of objects, including continuous system modeling,

discrete event system modeling, or hybrid system modeling. With the development of complex systems, the traditional modeling and simulation based on reductionism are challenging to work effectively because of their nonlinear properties, evolutionary uncertainty, and whole emergence. It is the new research orientation of high-performance simulation modeling theory and methods to integrate big data, machine learning, and high-performance modeling and simulation technologies and to replace reductionist decomposition modeling with “whole data” analysis [10, 11].

19.3.1.5 Challenges to Data-Driven Simulation Modeling Theory and Methods

The challenges to data-driven simulation theory and methods mainly include: (1) Studies focused on dynamic data-driven theory and methods, multi-disciplinary model interoperability theory and methods, digital twin theory and methods, and man-machine-material fusion modeling theory and methods, initiation of studies on intelligent data-driven simulation, and preliminary formation of real-time assisted decision-making theory and methods based on data-driven simulation for key application fields [11]. (2) The theory and methods of real-time assistant decision-making based on parallel simulation preliminarily formed for key application fields. Focus is placed on the theory and method of intelligent data-driven simulation and parallel systems, the theory and method of application-oriented data-driven automatic modeling and parallel execution, the automatic modeling theory and method of digital twin and man-machine environment integration system, the theory and method of parallel-degree enhancement for specific application fields, and the realization of real-time data-driven simulation-based assistant decision for various application fields. (3) Studies are focused on the ubiquitous simulation theory and method based on data-driven simulation and parallel system as well as on the digital twin theory and method, data-driven construction methods of the man-machine-material-environment integrated world, the parallel earth construction methods, and the formation of real-time assistant decision theory and methods based on data-driven simulation and parallel system for various fields [10].

19.3.1.6 Challenges to New AI-Based Simulation Modeling Theory and Methods

The new artificial intelligence system is a new intelligent interconnected system based on the new Internet and its combinations, which integrates man, machine, material, environment, and information within the information (cyber) space and physical space to provide intelligent resources and capabilities on demand by means of digital, networked, and intelligent technologies arising from new information and communication technology, new intelligent technology, and new professional technology in application fields (national economy, people’s livelihood, national security). Its system is characterized by the independent automation of perception, interconnection, collaboration, learning, analysis, cognition, decision-making, control and execution on the part of man, machine, material, environment, and information in the whole system throughout the whole life cycle. It implements the optimized integration of six factors (human, technology/equipment, management,

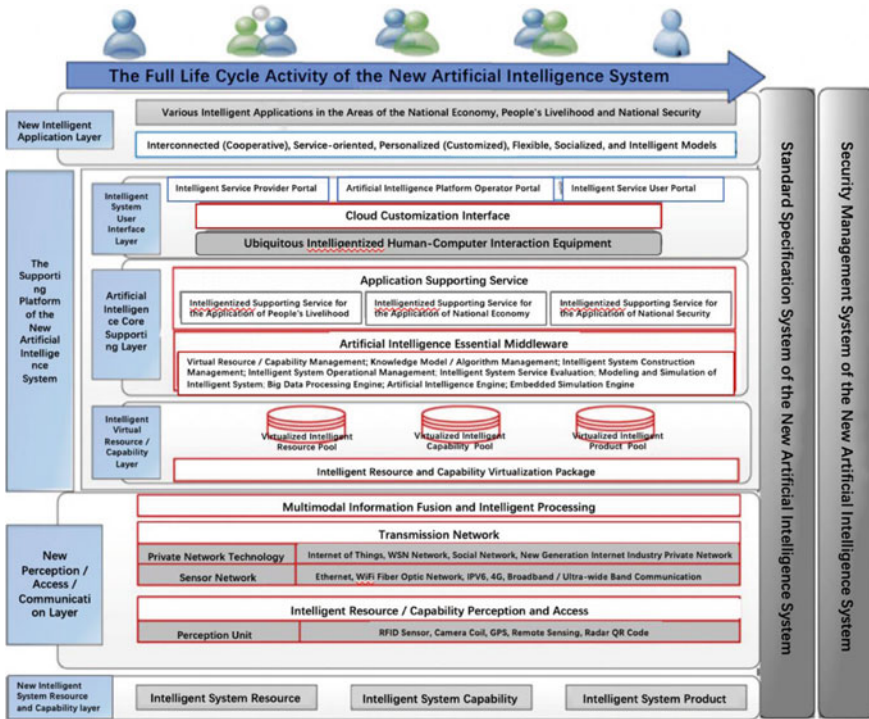


Fig. 19.2 Architecture of the new artificial intelligence system

data, material, capital) and six flows (human flow, technology flow, management flow, data flow, logistics flow, and capital flow) to form digital, networked and intelligent products, equipment/systems, and full life cycle activities. Its goal is to achieve an “innovative, green, open, shared, and personalized” society. The architecture of the new AI system is shown in Fig. 19.2 [2].

The challenges to new artificial intelligence-based simulation modeling theory and methods are mainly divided into two groups. The first group of challenge comes from object-oriented modeling (i.e., primary modeling), the challenges from new system modeling methods, especially those based on big data or deep learning, in view of such mechanisms, compositions, interactions, and behaviors as continuity, discreteness, qualitative/decision-making, and optimization with respect to man, machine, material, and environment in new artificial intelligence systems; the other group of challenge comes from modeling/algorithm (i.e., secondary modeling) on the simulator, that is, the challenges from developing new algorithms/methods for multi-level parallel and efficient simulation (such as efficient simulation operation of artificial intelligence system model) of various object-oriented models based on the architecture, software, and hardware characteristics of new AI systems and simulation systems [2].

19.3.1.7 Challenges to Simulation Modeling Theory and Method of System-of-Systems Confrontation

In future, the warfare will gradually develop toward intelligence and multi-domain, and be characterized as intelligent, systematic, unmanned, platform-based, and tactical, with the equipment system-of-systems confrontation increasingly intensified. It will pose a massive challenge to the traditional simulation technology for system-of-systems confrontation. The challenge to simulation modeling theory and method of system-of-systems confrontation covers two aspects: (1) design of component model based on the model framework. The modeler can complete the rapid construction of the model by combining the components. After model analysis, the simulation engine completes the comprehensive scheduling to obtain the simulation results. The most typical is the simulation model portability specification (SMP2.0). The design of the domain model framework is the key to the generalization of the equipment combat simulation model framework. It can be abstracted by modeling oriented to object, ontology, design pattern, and others and described with graphical specifications based on UML and other technologies. (2) Cognitive domain decision modeling framework based on rule bank. The core of intelligent modeling is cognitive domain decision modeling. The description of operational decisions is very complicated. The diversity of missions and the complexity of the space situation require sufficient flexibility in the description of operational decisions, while excessive flexibility may discourage modelers [12].

19.3.1.8 Challenges to the Unified Simulation Modeling Theory Based on DEVS

Simulation as a discipline is now in urgent need of a unified fundamental theory of simulation modeling, which should meet two basic requirements: First, it should have an excellent mathematical foundation, that is, it is based on formal mathematical theories; second, it should make the entire simulation knowledge systematic through the unified modeling theory. Discrete event system specification (DEVS) was proposed by Professor Zeigler of the University of Arizona in 1976. DEVS is based on strict mathematics, covering such types as distribution, parallel, real-time, and fuzziness. It unifies the descriptions for models, simulators, and experimental frameworks and includes theories, methods, and applications. It already has relatively independent and self-contained systems, theory, knowledge base, and methodology. Therefore, DEVS can be used as the basis or the starting point for establishing a unified modeling theory. It is a feasible way to improve the theoretical system of simulation and modeling based on DEVS. Similar to physics in pursuit of a grand unified theory, M&S also needs a unified modeling theory to unify various professional theories and provide the impetus for the development of the discipline. Even if DEVS cannot fully play its role as such a core theory, it also serves as an essential bridge to development in this direction. There are four basic models of DEVS: differential equations, difference equations, discrete events, and logical and symbolic forms. In recent years, various types of DEVS, together with the research findings based on the combination of DEVS and other modeling methods, show that DEVS is open and easy to integrate various theories and methods [13].

19.3.2 Challenges to Simulation Systems and Supporting Technology

19.3.2.1 Challenges to Virtual Reality System and Supporting Technology

In the context of the Internet of things, cloud computing, big data, smart city, etc., virtual technology needs to be more integrated with the Internet to open and integrate existing data and computing resources, express cognitive contents for users, and enable personalization on demand. In fact, it raises the technical requirements of virtuality-reality fusion. Only an excellent virtuality-reality fusion can produce a new networked virtual technology and application mode with a leap forward. At present, virtual reality technology is also integrating into the Internet, forming the mode of “Internet plus virtual reality.” It begins to integrate with cloud computing, big data, and mobile devices. As a result, higher requirements are raised to virtual reality technology from different aspects such as function and index. The current virtual reality technology can be combined with technologies related to vision, hearing, touch, and so on, but the intelligent part is still inadequate, leaving human-computer interaction unnatural. The development of new technologies in related fields may provide new opportunities and platforms for virtual reality technology. The challenge to the virtual reality system and supporting technology lies in integrating the existing data and resources with the development of related technologies and building more intelligent simulation systems. Virtual reality is closely related to human-computer interaction technology. The perception and cognition by cloud integration and human-computer interaction need to establish a human centered interaction paradigm, the key of which lies in the perception of multimodal information by cloud devices. After intelligent analysis and processing, such information is finally presented naturally, so that people can understand the connotation of big data more effectively. Deep fusion and processing of multimodal perceptual information can improve the accuracy and effectiveness of perception; the interaction may be made natural by barrier-free interaction means that breakthrough hardware constraints and meet people’s psychological and physiological requirements [10].

19.3.2.2 Challenges to Networked Simulation System and Supporting Technology

Challenges to networked simulation system and technology mainly cover: (1) Focusing on research on the domain simulation supporting technology based on cloud computing and establishing a network simulation environment in each application field for all kinds of simulation users. By taking the application field as the object, research is focused on the domain simulation technology based on cloud computing, the network simulation technology under the big data environment, and the network simulation technology based on artificial intelligence, and the simulation cloud-based network simulation environment is established in each application field to serve all-domain simulation users. (2) Focusing on research on the ubiquitous supporting technology for intelligent simulation and establishing a

network simulation environment with high intelligence in each application field. Research is focused on the new network-based ubiquitous simulation technology, distributed virtual reality technology, and large-scale parallel system technology. The networked simulation technology under the big data environment and the AI-based networked simulation technology are expanded to various simulation application fields, where a highly intelligent and user-friendly networked simulation environment is established. The effort is initiated to study the quantum computing simulation technology and the dynamic earth simulator construction and integration technology. (3) Focusing on the research of quantum computing simulation technology, computer-brain interaction simulation technology, and the implementation of all-round simulation of dynamic earth simulator. Research is expedited on ubiquitous simulation supporting technology based on the new network, brain-computer interconnection, Internet of everything, quantum computing, and big data, so as to build a dynamic earth simulator that can provide ubiquitous simulation capability, and to comprehensively realize ubiquitous multi-level simulation [10].

19.3.2.3 Challenges to Intelligent Simulation System and Technology

Challenges to intelligent simulation system and technology mainly cover: (1) development of intelligent simulation computer system for edge computing. It refers to the integrated intelligent simulation system that is aimed to optimize the overall performance of “system modeling, simulation operation, and result analysis/processing” and oriented to two kinds of simulation users (high-end modeling of complex systems and on-demand provision of high-performance simulation cloud service) and three kinds of simulation (mathematics, human in the loop, hardware in the loop/embedded simulation) by integrating emerging computer technology (such as cloud computing, Internet of things, big data, service computing, and edge computing), modern modeling and simulation technology, and supercomputer system technology. Its main research content involves computer system architecture, independent and controllable essential hardware/software, etc. (2) Development of cross-media intelligent visualization technology. It mainly includes the GPU group-based parallel visualization system technology and virtual reality fusion technology. The former involves data organization and scheduling technology of large-scale virtual scene, two-level parallel rendering technology based on multi-computer and multi-core technology, high-efficiency visualization technology of amorphous objects in a complex environment, and real-time dynamic global illumination technology. For example, the preliminary research results of the author’s team include: the GPU group-based parallel visualization system. (3) Development of the Internet-of-everything interface and intelligent special component technology. It mainly includes the research of CPS interface technology and the R&D of high-efficiency simulation acceleration components based on big data and artificial intelligence algorithm [10].

19.3.2.4 Challenges to High-Performance Simulation System and Supporting Technologies

Challenges to high-performance simulation system and supporting technologies mainly cover: (1) In view of the characteristics of complex system simulation, such as large sample, multi-entity, complex model, various data interaction, and in view of such requirements as domestic production, autonomous controllability, and multi-core + many-core, research is conducted by means of the deep integration of advanced software technology, artificial intelligence technology, and modeling and simulation technology, such as componentization, visualization, and automatic parallelization, focusing on development of the multi-level and multi-granular parallel supporting technology for high-performance simulation that can make full use of computing resources and provide an intuitive and easy-to-use development and operation platform for mining the parallelism of complex system simulation from the perspectives of sample, entity, model, algorithm, etc. (2) In view of the application demands by users for sharing simulation resources and capabilities on demand and obtaining simulation services anytime and anywhere, and based on cloud computing and essential big data services, research is focused on the intelligent service technology for lightweight container-based cloud simulation platform and the high-performance cloud simulation platform technology that supports efficient collaboration of generalized complex system simulation models to achieve efficient collaboration of simulation models in the cloud environment through the virtualization and service-oriented simulation system intelligence [10].

19.3.2.5 Challenges to Data-Driven Simulation System and Technology

Challenges to data-driven simulation system and technology mainly cover: (1) Realizing big data-driven simulation and modeling in the industry. It can implement the modeling of rapid environment, equipment, products, etc., and form a perfect simulation data governance mechanism through big data in the industry. Research is conducted on the construction technology for artificial systems, the computing experiment technology, and the parallel execution technology based on simulation cloud and big data, so as to establish a parallel simulation environment based on domain simulation cloud that meets the requirements of all-domain parallel simulation in typical application domains. (2) Realizing big data-driven simulation and modeling across domains. With data in different fields, models can be generated quickly for different scenes, and joint simulation of interdisciplinary and multi-disciplinary data can be supported; research is focused on the parallel simulation technology based on new network and artificial intelligence, the construction and experiment technology of large-scale comprehensive parallel systems, and the construction of a highly intelligent parallel simulation environment in the application fields to support real-time decision-making [11]. (3) Realizing ubiquitous data-driven simulation and modeling. The ubiquitous data extraction and processing are formed. According to the data, the relevant models are generated and simulated at any time. Research is focused on the parallel system technology based on quantum computing simulation technology and computer-brain interaction simulation technology so as to

build parallel earth. In addition, through the development of three stages, gradually developed are the security technology, security standards, and the security framework for data-driven simulation and parallel systems [10].

19.3.2.6 Challenges to New AI-Based Simulation Systems and Supporting Technologies

Major challenges come from the following five types of supporting technologies:

1. AI simulation cloud: Due to the distributed and heterogeneous characteristics of man, machine, material, and environment of the new AI system, it is necessary to virtualize and service all kinds of resources and capabilities so that users can obtain all kinds of resources and capabilities on demand, and then carry out various simulation activities such as mathematics, human in the loop, hardware in the loop/embedded simulation.
2. Intelligent virtual prototype engineering, including the support for heterogeneous integration and parallel simulation optimization of multi-disciplinary virtual prototype of complex objects in all kinds of new AI systems, and the optimized integration of human/organization, business management and technology, information flow, knowledge flow, control flow, and service flow in the whole system and life cycle.
3. Problem-oriented AI simulation language: For the modeling and simulation of new AI systems, a description language which is very similar to the original form of the system under study is used for input and then compiled to call the algorithm library, function library, and model library related to the field for high-performance simulation solution.
4. Construction of intelligent high-performance modeling and simulation system oriented to edge computing technology. Because some new artificial intelligence systems need simulation/calculation to be performed on the device terminals, such an intelligent system is required to ensure real-time operation. In addition, there is a potential need for collaborative solution and analysis between the high-performance computing center and massive front-end devices' intelligent computing capabilities.
5. Research on visualization technology based on cross-media intelligence, which is mainly intended for virtual scene computing and virtual reality fusion application in various new AI systems, by providing intelligent, high-performance, user-friendly visualization applications [2].

19.3.2.7 Challenges to Simulation Systems and Supporting Technologies of System-of-Systems Confrontation

Challenges to simulation system and supporting technologies of system-of-systems confrontation mainly include two aspects [12]:

1. A service-oriented confrontation simulation architecture is proposed to solve the following two core problems in simulation: The first is to upgrade the internal algorithm without changing the model interface and affecting the operation of the

simulation system, so as to ensure that the simulation system can obtain continuous model upgrade services. The second is to provide technical support for mode innovation of model management. The service-oriented architecture includes three types of roles: requester, provider, and registry. With the characteristics of loose coupling, it provides the maximum reuse degree for the model.

2. Building an living, virtual, constructive (LVC) universal platform based on real-time interaction technology. At present, one of the key development directions of system-of-systems confrontation simulation is the reuse, combinability, and interoperability of three kinds of heterogeneous simulation systems, so as to open up the three fields of simulation, experiment, and training and form the integrated simulation capability [12].

19.3.2.8 Challenges to DEVS-Based Simulation Interoperability Supporting Technology

In the early days, the development of M&S technology was mainly driven by its applications. With the development of the subject, theory, and method in the field of M&S will play an essential role in the development of technology. The breakthrough in the large-scale complex system simulation technology must be supported by theory and methods. Therefore, the merits and demerits of M&S theory are measured by judging whether it can promote the development of M&S technology, especially whether to break through the difficulties encountered in simulation practice, in addition to explaining M&S activities entirely and systematically. For the latter aspect, DEVS is not outstanding as yet, but it also provides a good foundation for the development of some critical technologies, such as interoperability, model verification and validation, and multi-resolution modeling. In terms of interoperability, DEVS can describe various types of models (discrete, continuous, real-time, intelligent, etc.) based on unified mathematics, which provides a better mathematical basis for model interoperability. DEVS also provides multi-faceted modeling, system entity structure/model base (SES/MB), and DEVS-based SOA, thereby delivering comprehensive support for interoperability. DEVS standardization is the key to achieving interoperability. Currently, such work as the establishment of the DEVS standard language and library, the standardization of DEVS modeling structure and semantics, the standardization of artificial intelligence DEVS models, the standardization of DEVS models for real-time systems, and the interoperability of DEVS tools at the model level and the determination of the DEVS kernels will promote the interoperability research in the field of simulation [13].

19.3.3 Challenges to Simulation Application Technology

19.3.3.1 Challenges to Virtual Reality Application Engineering

With the continuous development of virtual reality technology and applications, many practical application systems have been built in the fields of military, public security, industrial design and large-scale engineering, medicine, and cultural education. As virtual technology is commercialized, there appear more and more virtual

experience applications worthy of promotion, such as a virtual wedding, a virtual family gathering, and a virtual company meeting. At present, the foreseeable application fields of virtual technology include: transportation, health care, emergency rescue, environmental pollution, natural disasters, population education, intelligent manufacturing, smart town, cultural creativity, translational medicine, e-commerce and other industries, as well as online services, social management, and so on. At present, the equipment is developing toward lightweight and portability. Now, most of the supporting technologies are available. The development roadmap for virtual technology and application simulation application engineering is based on the needs of various industries, focusing on user experience, and promoting the implementation of relevant applications through market demands [10].

19.3.3.2 Challenges to Networked Simulation Application Engineering

Challenges to networked simulation application engineering mainly cover: (1) Focusing on the construction of domain simulation resources, carrying out the construction of domain simulation model engineering, domain simulation data engineering, domain ontology engineering and domain high-performance simulation algorithm library, and establishing domain simulation cloud to provide Web-based services of simulation models, algorithms and tools for economic, social, and national defense fields. (2) Focusing on the construction of virtualization, integration and sharing of domain simulation resources, carrying out intelligent domain simulation engineering, and building intelligent domain simulation cloud, a visualized, intelligent, multi-functional cloud for real-time socio-economic simulation and providing cloud simulation services for simulation application systems. Based on the domain simulation cloud, a comprehensive parallel system is built for the simulation applications in the fields of economy, society, and national defense. (3) Focusing on the construction of dynamic earth simulator, providing universal simulation services to protein folding of molecular dynamics, weather forecasting, drug research, development, space exploration, nuclear simulation, so as to realize the synchronous operation of the simulation model with the actual system. The part people pay attention to in the real world has a digital twin in the dynamic earth simulator, where virtual entities interact with physical entities, and any question on issues of common concern raised by people from time to time may be answered by the model quickly with accurate explanations and potential exploration of the possible development of events at different granularity [10].

19.3.3.3 Challenges to Intelligent Simulation Application Engineering

Challenges to intelligent simulation application engineering mainly cover:

1. Fully developing software definition network (SDN), using the big data collected by the SDN controller for deep learning to improve the reliability of VV&A.

2. Further improving intelligent analysis of big data, especially collection, classification, identification, and comprehensive processing of simulation experiment data closely related to intelligent simulation.
3. Using new artificial intelligence technology to realize full life cycle VV&A, full system VV&A, hierarchical VV&A, all-personnel VV&A, and all-round management VV&A.
4. Realizing intelligent data acquisition, zero waiting for data transmission, data storage learning, and brain-like data analysis in simulation experiment [10].

19.3.3.4 Challenges to High-Performance Simulation Application Engineering

Challenges to high-performance simulation application engineering mainly cover:

1. Intelligent VV&A of simulation models. The credibility of modeling and simulation is the basis of its survival, while the biggest problem of modeling and simulation is the verification of its credibility, especially the simulation model of the complex system. Verification and validation become a significant problem in simulation currently. The arrival of the big data era provides a new solution for VV&A of the complex system simulation models, which verifies not only the authenticity of the simulation but also the correctness of each model through fidelity and credibility data of simulation that are obtained by using sufficient similar cases.
2. Multi-sample simulation experiment design technology based on big data and artificial intelligence. In view of the characteristics of large-scale complex simulation system such as complex spatial structure, numerous schemes, and many types of factors affecting system efficiency and complex relationships, research is focused on the intelligent simulation test design technology for complex large systems based on big data and artificial intelligence, and the new efficient means for the simulation test design of complex large-scale systems based on the information mined by big data and using artificial intelligence technology to guide the search process of sample space.
3. Big data mining and intelligent evaluation technology for simulation results. In view of the complicated relationship between simulation factors and performance indicators in complex system simulation, research is focused on the intelligent simulation evaluation technology based on big data and deep learning to support big data analysis and intelligent evaluation of performance indicators and simulation factors [10].

19.3.3.5 Challenges to Data-Driven Simulation Application Engineering

Challenges to data-driven simulation application engineering mainly cover:

1. Focusing on domain dynamic data-driven and parallel system engineering. Based on the construction of domain model engineering, domain simulation data engineering, domain ontology engineering, and domain high-performance

simulation algorithm library, domain dynamic data-driven parallel system engineering is developed based on the domain simulation cloud to provide support for real-time assistant decision-making in economic, social, and national defense fields [11].

2. Focusing on dynamic data-driven and parallel system engineering in the field of intelligence. Based on automatic interaction of virtual-real systems, the domain intelligent dynamic data drive and parallel system engineering are developed based on the intelligent domain simulation cloud to improve the parallel degree of the parallel system and provide high-quality support for real-time assistant decision-making in the fields of economy, society, and national defense.
3. Focusing on the construction of parallel earth engineering and providing ubiquitous simulation services in cooperation with the dynamic earth simulator engineering [10].

19.3.3.6 Challenges to New AI-Based Simulation Application Engineering

The challenges to new AI-based simulation application engineering cover mainly:

1. Verification, validation, and acceptance (VV&A) methods for intelligent models, including primary model approval, execution accuracy approval of simulation system algorithm, and user approval of simulation execution results.
2. Management, analysis, and evaluation of intelligent simulation experiment results. It is necessary to conduct efficient concurrent simulation and efficient acquisition, management, and analysis of simulation results since a large number of demands for intelligent simulation applications come from the rapid simulation and prediction of the possible behavior mode and performance of the whole system.
3. Big data-based intelligent analysis and evaluation technology. The application of new artificial intelligence systems is necessary to consider the complexity of the actual man, machine, material and environment, as well as the constraints of various running equipment. It is also necessary to consider such application modes and technologies as big data access and storage management, big data cloudization, big data analysis and decision-making, big data visualization, and result evaluation [14].

19.3.3.7 Challenges to Simulation Application Engineering of System-of-Systems Confrontation

With the continuous updating of warfare equipment, the rapid development of technology, and the continuous changes in combat modes, future operations will gradually grow from mechanization to informationization and intelligence. Future warfare will gradually enter a period in which “in-depth practice of information warfare” and “initial development of intelligent warfare,” and warfare will gradually evolve from land, air, and naval battles to the six-dimensional integration of land, sea, air, space, power, and network. The “Multi-domain Battle” by the U.S. Army, the “AirSea Battle” by the U.S. Air Force, and the “Distributed Lethality” by

the U.S. Navy are all operational concepts aimed to address all-domain operations, with main features including: Integrated joint operations, multi-dimensional cooperative operations, and unmanned intelligent operations. The new global combat mode has profoundly affected the combat methods of weapons and equipment, which is manifested as the intensification of equipment system confrontation. The traditional “one-on-one” confrontation mode is now difficult to come up with new requirements. As the system-of-systems confrontation grows more intensive and longer with several means, a significant challenge is posed to the traditional confrontation simulation technology. In the case where actual confrontation is challenging to carry out, a new confrontation simulation technology is urgently needed to address the new system-of-systems confrontation. In future, the warfare will gradually develop toward intelligence and multi-domain, and be characterized as intelligent, systematic, unmanned, platform-based and tactical, with the equipment system-of-systems confrontation increasingly intensified. It will pose a considerable challenge to the traditional simulation technology for system-of-systems confrontation, including the adaptation of the combat system architecture to dynamic changes, the rapid formation of model system simulation under the system-of-systems operation, the synchronization of modeling technology to the pace of intelligent operations, and the adaptation of the combat system-of-systems simulation platform to multi-resolution application requirements, etc. [12].

19.3.3.8 Challenges to DEVS-Based Simulation Application Engineering

Due to the modular and hierarchical modeling and the analysis capability based on simulation, DEVS formal mechanism and its various extension forms have been used in many application engineering and scientific fields. The application fields of DEVS include: manufacturing system, military system and infrastructure, real-time embedded system, transportation and traffic system, robot and multi-robot system, agent M&S, ecological and environmental system, biological system and social system, etc. DEVS continues to expand in terms of scope, depth, and breadth of its use. For example, DEVS is a significant candidate for simulation-based acquisition (SBA) simulation model description standards. By using DEVS-based interoperability standards, the M&S system and C2 system are integrated to support C2 system decision-making and test evaluation. DEVS is also used as the whole framework of knowledge-based control for steel production process control. Because DEVS delivers a good formal description, it is easy to be used in the automatic generation of simulation application test system, such as the automatic test of Link 16 using DEVS and XML [13].

The capability maturity model (CMM) for software development plays a crucial role in ensuring the success of software projects. CMM and CMM integration (CMMI) originated from software engineering, but they have been applied in many other fields for years. However, in M&S, this standardized and systematic evaluation method have not been developed for M&S processes [15]. Based on CMM/CMMI, the capability maturity model (MS-CMMI) for modeling and simulation process can be established by the following methods: By analyzing the

characteristics of modeling process and complex system simulation, the similarities and differences between modeling/simulation process and software development process are found out, and then indexes and metrics are defined for M&S process. An MS-CMMI evaluation system (evaluation methods, standards, tools, organizations, etc.) is set up to evaluate the structured capability of model developers or model users (using models for simulation).

References

1. Government Office for Science (2018) Computational modelling: technological futures
2. Li BH, Chai XD, Zhang L, Li T, Qin DZ, Lin TY, Liu Y (2018) Preliminary study of modeling and simulation technology oriented to neo-type artificial intelligent systems. *J Syst Simul* 30(2):349–362
3. Zhang L, Zhou LF, Ren L, Laili YJ (2019) Modeling and simulation in intelligent manufacturing. *Comput Ind* 112
4. Zhang L, Shen Y, Zhang X, Song X, Tao F, Liu Y (2014) The model engineering for complex system simulation. In: *The 26th European modeling and simulation symposium (simulation in industry)*, Bordeaux, 10–12 Sept
5. Tuegel EJ, Ingraffea AR, Eason TG, Spottswood SM (2011) Reengineering aircraft structural life prediction using a digital twin. *Int J Aerospace Eng*
6. Tao F, Cheng JF, Qi QL et al (2017) Digital twin driven product design, manufacturing and service with big data. *Int J Adv Manufact Technol*
7. Park J (2008) Augmented reality based re-formable mock-up for design evaluation. In: *IEEE International symposium on ubiquitous virtual reality*, July 2008. ISUVR 2008, pp 17–20
8. Ng LX, Oon SW, Ong SK, Nee AY (2011) GARDE: a gesture-based augmented reality design evaluation system. *Int J Interact Des Manufact (IJIDeM)* 5(2):85
9. Pan YH (2017) New generation of the Chinese artificial intelligence. In: *The 1st world intelligence conference*, Tianjing
10. China Simulation Federation (2018) Development trend forecast and roadmap of simulation science and technology (internal report)
11. World Technology Evaluation Center, Inc. (2009) Panel report on international assessment of research and development in simulation-based engineering and science
12. Bi CJ (2014) Modeling and simulation are geared to challenge in the age of big data. *Compu Simul* 31(1):1–3
13. Qiu XG, Duan W (2009) On research development of DEVS and its function to modeling and simulation discipline. *J Syst Simul* 21(21):31–38+43
14. Li BH et al (2019) Research on modeling and simulation technology in the era of ‘Intelligence +’
15. Fujimoto R, Bock C, Chen W, Page E, Panchal JH (2017) *Research challenges in modeling and simulation for engineering complex systems*. Springer

Appendix A: Terminology and Other Reference Documents

Compiled by Tuncer Ören

Terminology

Establishing a common set of concepts and describing terms is pivotal for every discipline. There will always be variation used by communities of interest or community of practice, but a common core vocabulary allows to communicate new ideas. The following two seminal papers are highly recommended to be studied to set the foundation for the terminology used within the Body of Knowledge for M&S:

Ören, T.I. (2011a). The Many Facets of Simulation through a Collection of about 100 Definitions. *SCS M&S Magazine*, 2:2 (April), pp. 82–92. <http://scs.org/wp-content/uploads/2016/12/2011-04-Issue06-6.pdf> (Accessed: 2021-06-30)

Ören, T.I. (2011b). A Critical Review of Definitions and About 400 Types of Modeling and Simulation. *SCS M&S Magazine*, 2:3 (July), pp. 142–151. <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.371.4566&rep=rep1&type=pdf> (Accessed: 2021-06-30)

Furthermore, this set of appendices provides additional sources:

- A1: Glossary of Terms Used in this Document
- A2: Dictionaries of Modeling and Simulation (M&S), Cybernetics, and Systems
- A3: M&S Associations/Organizations/Committees/Groups/Research Centers
- A4: Additional Resources.

A1 Glossary of Terms Used in This Document

The definitions are from the author(s) of the sections associated to each definition.

In cases where section numbers are omitted, the names indicate who provided the definitions.

	In section
A	
An abstract DEVS simulator describes implementable simulators of DEVS model	Section 1.4.6
Abstraction is the process of constructing a lumped model from a base model intended to be valid for the real system in a given experimental frame	Section 1.4.6
Acceptor monitors an experiment to see the desired experimental conditions are met	Section 1.4.2
Advisory professional codes for simulationists provide guidelines that help the simulationist to make good decisions, often very similar to Code of Best Practices	Section 8.2.5
Augmented reality (AR for short) is a new technology that integrates the real-world information and the virtual world information seamlessly. Through computer technology, it simulates and superposes physical information (visual information, sound, taste, touch, etc.) that is difficult to experience in a certain spatio-temporal range of the real world	
B	
A base model is valid within a larger set of experimental frames (with respect to a real system) than the lumped model	Section 1.4.4
C	
Complexity of a model can be measured by the resources required by a particular simulator to correctly interpret it	Section 1.4.4
A conceptual model describes, at the highest level of abstraction, the general knowledge about the system under study to highlight system entities (or classes), as well as the relationships between them	Section 2.3
Computer simulation: Simulation where the computations are performed on a computer	Ören
Computer simulation: Simulation of computers (similar to simulation of traffic is called traffic simulation) Also synonym of computerized simulation	Ören
Computerized simulation: Simulation study where all knowledge processing are done on a computer	Ören
Constraint (or conservative) model provides system descriptions in terms of laws and constraints. Differential equations or difference equations are examples of adequate formalism for this abstraction level	Section 2.3
D	
Data (To see definitions of 64 types of data in Sect. 2.2)	Section 2.2
Data level (or level 1 system specification) is a data base of measurements and observations made for the source system	Section 1.4.1
Data poor environment is an environment where historical data lacks desirable qualities (such as quality or quantity)	Section 1.4.2
Data-rich environment is an environment where data is abundant from prior experimentation or can easily be obtained from measurements	Section 1.4.2
De Facto standard: A standard widely used, but not endorsed by a standards organization	Section 6.1.1

(continued)

(continued)

	In section
De Jure standard: A standard endorsed by a standards organization	Section 6.1.1
A declarative model provides system description in terms of sequences of events that move the system from one state to another	Section 2.3
Descriptive model analysis —Syn. model characterization	
Discrete Event System Specification (DEVS)	Section 1.4.2
A DEVS model is a state machine with time included as a component of the state	Section 3.4
A DEVS model can be regarded as a state machine that specifies transitions of the total state $[s, \Delta t_e]$ —that is, transitions of the state and elapsed duration in combination—rather than transitions of the state s by itself	
Digital simulation: A simulation where the computation is carried out on a digital computer	Ören
Disciplinary professional codes impose negative consequences for violations of standards	Section 8.2.5
A domain rule represents some domain-dependent state transformation mechanisms	Section 2.4.5
E	
Epistemology is the branch of philosophy that copes with the question of how to gain new knowledge within a discipline	Section 15.1
Ethics is the branch of philosophy that studies moral problems, that is, problems of right and wrong action.	Section 8.1
Ethics , also called moral philosophy, is the discipline concerned with what is morally good and bad and morally right and wrong. The term is also applied to any system or theory of moral values or principles.	Section 8.2.1
Evaluative model analysis —Syn. model evaluation	Section 2.1
Experimental frame is a specification of the conditions under which system is observed or experimented with	Section 1.4.2
F	
A functional model provides system descriptions in terms of sequences of functions (the outputs of the ones feeding the inputs of the others), arranged such that the final treatment performed by the system is found downstream of this sequence	Section 2.3
G	
Gaming simulation provides experience for entertainment and for training purposes	Section 1.1
General domain model	
Generator generates input segments to the system	Section 1.4.2
Grading professional maturity involves assessing processes used in a profession, individuals and organizations active in this profession, and products and/or services of the profession	Chapter 10
H	
Haptic feedback technology is a technology that is utilized to reproduce real tactile sensation for users through a series of actions such as force or vibration provided by the professional equipment	Section 14.2.3

(continued)

(continued)	
	In section
High-fidelity model may refer to a model that is both high in detail and in validity (in some understood experimental frame)	Section 1.4.3
High-performance simulation is a new simulation paradigm, new approach, and new ecosystem, which combines four kinds of technologies including emerging computer science technology (i.e., cloud computing, IoT, big data, service computing, and edge computing), modern modeling and simulation technology, supercomputer system technology, and artificial intelligence technology, aimed to optimize the overall performance of system modeling, simulation operation and result analysis/processing for 2 kinds of users (high-end simulation user and massive simulation user group) and 3 types of simulations (constructive simulation, virtual simulation, and live simulation)	Section 11.6
Hybrid simulation is the combined application of simulation techniques like system dynamics (SD), discrete event simulation (DES), and agent-based simulation (ABS) in the context of a single simulation study	Section 3.6.2
Hybrid simulation: A simulation where the computation is carried out on a hybrid computer	Ören
L	
Logical time is measured by ticks of a clock somehow embedded in a model	Section 1.4.5
M	
Measures of effectiveness measure how well the overall system goals are being achieved	Section 1.4.2
Measures of performance are output variables that typically judge how well parts of a system are operating	Section 1.4.2
A meta-model performs some domain-independent functionalities/states/activities	Section 2.4.5
Metric time —Syn. physical time	Section 1.4.5
Model is a representation of a simulant, broadly grouped into conceptual and executable types	Section 1.4.6
(A model can be conceived as any physical, mathematical, or logical representation of a system, entity, phenomenon, or process.)	Section 1.4.2
Each model is a simplification and abstraction of reality, and the model becomes the reality of the simulation	Section 15.2
Model analysis consists of descriptive model analysis (model characterization) and evaluative model analysis (model evaluation)	Section 2.1
Model base management includes model search, semantic model search, and model integrity	Chapter 2
Model characterization (or descriptive model analysis) consists of model comprehensibility (model documentation and model ventilation) and model usability	Section 2.1
Model engineering (ME) or named as model lifecycle engineering (MLE) deals with the model lifecycle credibility and scalability problem for complex systems, especially systems-of-systems	Section 2.4
Model engineering is a general term for theories, methods, technologies, standards, and tools relevant to a systematic, standardized, quantifiable engineering methodology that guarantees the credibility of the full lifecycle of a model with the minimum cost	Section 2.4.2

(continued)

(continued)

	In section
Model lifecycle contains six phases, i.e., problem definition, model design, model construction, verification, validation, and accreditation (VV&A), model implementation, model evolution and reconfiguration, and model maintenance	Section 2.4.1
Model Lifecycle Engineering —Syn. model engineering	Section 2.4
Model evaluation or evaluative model analysis can be done with respect to modeling formalisms, another model (model comparison), real system, and goal of study	Section 2.1
Model processing consists of model analysis, model transformation, and behavior generation	Section 2.1
Model realism (or model veracity, model verisimilitude) consists of adequacy of model structure (static structure (relevant variables, interface of models) and dynamic structure) and adequacy of model constants and parameters	Section 2.1
Model reconfiguration means to change part of model during its runtime	Section 2.4.6
Model ventilation is examination of its assumptions, deficiencies, limitations, etc	Section 2.1
Modeling is a task-driven purposeful simplification and abstraction of a perception of reality	Section 16.2
Modeling is creating a simplified representation of something	Section 18.1
Monte Carlo Simulation —synonym: stochastic simulation	Ören
Monte Carlo simulation is use of random variates to solve deterministic problems such as computation of multi-dimensional integrals	Ören
Moral philosophy —Syn. ethics	
Morals are the practice of ethics	Section 8.2.1
N	
Numerical simulation designates the process by which a program is executed on a (digital) computer in order to represent a given phenomenon	Section 11.1
O	
Outcome measures are measures of the effectiveness of a system in accomplishing its goal which are required to evaluate the design alternatives	Section 1.4.2
Output variables are variables of a model whose values are computed during execution runs of the model in order to compute outcome measures	Section 1.4.2
P	
Physical time , also called metric time or wall-clock time, is measured by ticks of physical clocks	Section 1.4.5
Predictive validity of a model necessitates not only replicative validity, but also the ability to predict as yet unseen system behavior	Section 1.4.3
Proprietary standard : A standard that belongs to an entity that exercises control over the standard	Section 6.1.1

(continued)

(continued)	
	In section
R	
Reliable model: If the simulation model is reliable, conclusions and perceptions obtained from its execution can be reliably used to draw inferences about the simuland	Section 7.2
Replicative validity of a model is affirmed if, for all the experiments possible within the experimental frame, the behavior of the model and system agree within acceptable tolerance	Section 1.4.3
The results of simulation are the output produced by a model during a simulation	Section 1.4.6
S	
A scientific explanation is a deductive argument from a general law and certain other premises to the desired conclusion. The general law needs to be an essential premise. Deductively invalid arguments fail to explain, as do arguments not containing a general law	Section 16.3
A simuland is the real-world system of interest. It is the object, process, or phenomena to be simulated	Section 1.4.6
Simulation: (Please see: Appendices A2 and A3): A2—Ören, T.I. (2011a). The Many Facets of Simulation through a Collection of about 100 Definitions. SCS M&S Magazine, 2:2 (April), pp. 82–92 A3—Ören, T.I. (2011b). A Critical Review of Definitions and About 400 Types of Modeling and Simulation. SCS M&S Magazine, 2:3 (July), pp. 142–151	Ören
Simulation (<i>experimentation aspect</i>): Simulation is performing goal directed experiments with models of dynamic systems	Section 1.1
Simulation (<i>experience aspect</i>): Simulation is providing experience under controlled conditions for: 1. <i>Training</i> , i.e., for gaining/enhancing competence in one of the three types of skills: (a) motor skills (virtual simulation or use of simulators), (b) decision and/or communication skills (constructive simulation such as business games, war games, or peace games; aka serious games), (c) operational skills (live simulation) 2. For <i>entertainment purposes</i>	Section 1.1
Simulation model is a set of instructions, rules, equations, or constraints for generating I/O behavior. In other words, a simulation model is a specification with state transition and output generation mechanisms to accept input trajectories and generate output trajectories depending on its initial state setting	
Simulator is a computational device for generating behavior of the model	
A simulator is any computation system (such as a single processor, a processor network, the human mind, or more abstractly an algorithm), capable of executing a model to generate its behavior	
Simulator correctness: A <i>simulator correctly simulates a model</i> if it is guaranteed to faithfully generate the model's output trajectory given its initial state and its input trajectory	

(continued)

(continued)	
	In section
Source level (or level 0 system specification) identifies a portion of the real world that we wish to model and the means by which we are going to observe it	Section 1.4.1
Structural validity of a model necessitates that the model not only is capable of replicating the data observed from the system but also mimics in step-by-step, component-by-component fashion, the way that the system does its transitions	Section 1.4.3
Structural validity of a model necessitates that the model not only is capable of replicating the data observed from the system but also mimics in step-by-step, component-by-component fashion, the way that the system does its transitions	Section 1.4.3
System (or source system) is the real or virtual environment that we are interested in modeling. It is viewed as a <i>source of observable data</i> , in the form of time- indexed trajectories of variables	Section 1.4.2
System behavior database is the data that has been gathered from observing or otherwise experimenting with a system	Section 1.4.2
System integration technology refers to the technology that can help integrate various technologies applied in VR/AR (such as information synchronization technology, model calibration technology, data conversion technology, recognition and synthesis technology)	Section 14.2.1
T	
Time base is an ordered set, usually the real numbers, for indexing events that model the flow of actual time	Section 1.4.5
Transducer that observes and analyzes the system output segments	Section 1.4.2
V	
Validation is the process of determining if a model behaves with satisfactory accuracy consistent with the study objectives within its domain of applicability to the simuland it represents	Section 1.4.6
Validity of a model is the assurance that the model faithfully captures the system behavior only to the extent demanded by the objectives of the simulation study. The concept of validity answers the question of whether it is impossible to distinguish the model and system in the experimental frame of interest	Section 1.4.3
Verification is the process of determining if an implemented model is consistent with its specification	Section 1.4.6
Virtual reality (VR for short) is the use of computer simulation to generate a three-dimensional virtual world, providing users with simulated vision, hearing, touch, and other senses so that users feel as if they were in the real environment and were able to observe things in the three-dimensional space immediately without any restrictions	Section 14.2.1
W	
Wall-clock time —Syn. physical time	Section 1.4.5

A2 Dictionaries of Modeling and Simulation (M&S), Cybernetics, and Systems

Modeling and simulation (29 dictionaries)	
Augmented reality	
<u>AR BimDictionary</u>	Augmented reality simulation (AR)
Analog and hybrid computers	
<u>NEES</u>	Network for Earthquake Engineering Simulation (NEES) Hybrid Simulation Primer and Dictionary
<u>SCS-1975</u>	The Society for Computer Simulation—Definitions of Terms for Analog and Hybrid Computers (1 November 1975). <i>Simulation</i> , 26:3 (March 1976), 80–86
Biology	
<u>SB</u>	Biology Online Dictionary
Education	
<u>UCF</u>	UCF (University of Central Florida) Modeling and Simulation Terminology (Modeling and Simulation for Instructional Design)
Environmental engineering	
<u>EnvEng</u>	General Glossary of Terms relating to Modelling and Simulation for Environmental Engineering
<u>Flight simulation</u>	Definitions and Abbreviations for Flight Simulation Training Devices
Gaming	
<u>GameDeveloper</u>	Dan Carreker (2012). <u>The Game Developer’s Dictionary: A Multidisciplinary Lexicon for Professionals and Students</u> . Course Technology Press, Boston, MA, USA
G.I. Gibbs	G.I. Gibbs (1978). <i>Dictionary of Gaming, Modelling & Simulation</i> . E & F N Spon (An imprint of Routledge) (at Amazon)
<u>L. Pulsipher</u>	Glossary for Game Designers. Copyright 2010 Lewis Pulsipher
<u>Sloper</u>	FAQ 28: A Glossary of Game Biz terms
General M&S	
<u>ACM SIGSIM M&S</u>	ACM SIGSIM Modeling and Simulation Glossary
<u>IEEE</u>	IEEE Standard Glossary of Modeling and Simulation Terminology (IEEE 610.3–1989) withdrawn
<u>Li, Bo Hu et al.</u>	Li, Bo H, T.I. Ören, Qingping Zhao, Tianyuan Xiao, Zongji Chen, and Guanghong Gong et al. (2012). <i>Modeling and Simulation Dictionary: Chinese-English, English-Chinese (over 9000 terms), Chinese Science Press, Beijing, P.R. of China. (With the contribution of 30 Chinese colleagues—contributors) (front cover)</i>
Li, Bo Hu et al. (2nd ed.)	Li, Bo Hu et al. (2019). 2nd edition of the <i>Modeling and Simulation Dictionary: Chinese- English, English-Chinese (about 12 500 English terms)</i> . Chinese Science Press, Beijing, P.R. of China
<u>Ören et al.</u>	Ören, T.I. and The French Team: (2006). <i>Modeling and Simulation Dictionary: English-French-Turkish</i> . Marseille, France. With the support of Centre Nationale de La Recherche Scientifique, I3, Université Paul Cézanne—Aix-Marseille III, and LSIS (286 pp). ISBN: 2-9524747-0-2

(continued)

(continued)

Modeling and simulation (29 dictionaries)**Augmented reality**

Currently, there are over 12 500 English terms in the master file. The work can be continued to finalize English-French, English-Italian, English-Spanish, and English-Turkish M&S dictionaries

HealthcareHealthcare

Lioce, L. (ed.). Healthcare Simulation Dictionary (2nd edition). 2020. AHRQ Publication No. 20-0019

INALCSL

INACSL Standards Committee (2016, December). INACSL standards of best practice: SimulationSM Simulation glossary. Clinical Simulation in Nursing, 12(S), S39-S47. h

SSH

Glossary of Terms Abridged from SSH Accreditation Manual

VHA SimLEARN

Clinical Simulation and Resuscitation Glossary of Training Terms

LightingLDG

Lighting Design and Simulation Terminology

Military SimulationDoD M&S glossary

DoD M&S Glossary—DoD 5000.59-M

SEDRIS

SEDRIS (Synthetic Environment Data Representation and Interchange Specification) Glossary (2007)

SISO-Fidelity

SISO Glossary of Terms Applied to Fidelity

SISO_DIS

Distributed Interactive Simulation

UK_SEMS

UK Ministry of Defence Synthetic Environment & Modelling & Simulation (SEMS) Glossary

Wikis

Glossary of military modeling and simulation: Wikis

Verification and validationNASA

Glossary of Verification and Validation Terms

NPARC V&V

Glossary of Verification and Validation Terms

Cybernetics and Systems (5 dictionaries)F. Heylighen

Web Dictionary of Cybernetics and Systems (Principa Cybemetica Web)

IECaS

International Encyclopedia of Cybernetics and Systems, edited by Charles François, (1997) K. G. Saur publishing, München, Germany (second ed. 2004)

ST

Glossary of Systems Theory (Wikipedia)

TSE

List of Types of Systems Engineering (Wikipedia)

TST

List of Types of Systems Theory (Wikipedia)

A3 M&S Associations/Organizations/Committees/Groups/Research Centers

Index

High Level Recognition of M&S (3)

Networking of Professional Organizations

Medical Organizations (10)

Others (19)

Associations

International Associations/Groups (31)

by Country Associations (62)

by Region/Language (22)

Research Centers/Groups (40)

Military Organizations

NATO (4)

Europe (1)

Country (28).

High-Level Recognition of M&S

- USA—H. Res. 487—Recognizing the contribution of modeling and simulation technology to the security and prosperity of the United States and recognizing modeling and simulation as a National Critical Technology. 110th Congress (2007–2008) <https://www.congress.gov/bill/110th-congress/house-resolution/487>
- USA—H. Res. 855—Enhancing Safety in Medicine Utilizing Leading Advanced Simulation Technologies to Improve Outcomes Now Act of 2009. 111th Congress (2009–2010) <https://www.congress.gov/bill/111th-congress/house-bill/855/text>
- USA—Congressional Modeling and Simulation Caucus <https://www.trainingsystems.org/initiatives/advocacy/congressional-modeling-and-simulation-caucus>.

Networking of Professional Organizations

Medical Simulation

AIMS—Advanced Initiative in Medical Simulation

ASPE—The Global Network for Human Simulation Education

Associations for Simulation in Healthcare

CSA—The California Simulation Alliance

Global Simulation Societies—Simulation Societies Around the World

GNSH—The Global network for Simulation in Healthcare

SSH Affiliates

SUN—Simulation User Network (Medical, Nursing, and Healthcare)

World Medical Simulation Centre Database.

Others

- AMSC—Alabama Modeling & Simulation Council
- CSSS—Czech and Slovak Simulation Society
- DBSS—Dutch Benelux Simulation Society
- ETSA—European Training and Simulation Association
- EUROSIM—Federation of European Simulation Societies
- G.A.M.E.S. Synergy Summit (**G**overnment, **A**cademic, **M**ilitary, **E**ntertainment and **S**imulation)
- IMSF—International Marine Simulator Forum
- ITSA—International Training and Simulation Alliance
- M&SNet—McLeod Modeling & Simulation Network (of SCS)
- MISS—McLeod Institute of Simulation Sciences (of SCS)
- MSLS—M&S Leadership Summit
- NCS—The National Center for Simulation (USA)
- NEMSC—New England Modeling & Simulation Consortium
- NMSC—National Modeling & Simulation Coalition
- NTSA—National Training Systems Association (USA)
- SAGSAGA—Swiss Austrian German Simulation and Gaming Association
- SIM Center Directory
- SIMS—Scandinavian Simulation Society
- WSA—Worldwide Simulation Alliance.

Associations—International

- ABSEL—Association for Business Simulation and Experiential Learning
- ACM SIGSIM—ACM Special Interest Group on Simulation
- AISB—The Society for the Study of Artificial Intelligence and the Simulation of Behaviour
- AMSE—Association for the Advancement of Modelling and Simulation Techniques in Enterprises
- ANGILS—Alliance for New Generation Interactive Leisure and Simulation
- DIGRA—Digital Games Research Association
- EBEA—The Economics and Business Education Association
- IASTED—International Association of Science and Technology for Development
- IBPSA—International Building Performance Simulation Association
- IGDA—International Game Developers Association
- IMA—International Microsimulation Association
- IMACS—International Association for Mathematics and Computers in Simulation
- IMUNA—Education Through Simulation
- INACSL—International Nursing Association for Clinical Simulation and Learning
- INFORMS Simulation Society
- IPSS—International Pediatric Simulation Society

- ISAGA—International Simulation and Gaming Association
- ISHS—International Society for Human Simulation
- M&SPCC—Modeling and Simulation Professional Certification Commission
- Modelica—Modelica Association
- MSRC—Maritime Simulation and Resource Center
- NAFEMS—The International Association for the Engineering Modelling, Analysis and Simulation Community
- NASAGA—North American Simulation and Gaming Association
- PSE—Professional Simulation Engineer Certification
- SAE—Human Biomechanics and Simulation Standardization Committee
- SAGSET—The Society for the Advancement of Games and Simulations in Education and Training
- SCS—Society for Modeling & Simulation International (Formerly Society for Computer Simulation)
- SGA—Serious Games Association
- SGI—Serious Games Initiative
- SSH—Society for Simulation in Healthcare
- SSSG—Social Simulation and Serious Games (SIG of ESSA).

Associations—By Country

- **Albania**
ALBSIM—Albanian Simulation Society
- **Australia**
ASSH—The Australian Society for Simulation in Healthcare
SimAust—Simulation Australia
- **Belgium**
FRANCOSIM—Société francophone de simulation
- **Bulgaria**
BASAGA—Bulgarian Academic Simulation and Gaming Association
Bulsim—Bulgarian Modeling and Simulation Association
- **Canada**
Can Sim—Canadian Alliance of Nurse educators using Simulation
Sim Center Canada
SimGhosts—Simulation Canada Affiliation
Simulation Canada
- **China**
CASS—Chinese Association of System Simulation
SASS—Shanghai Association for System Simulation (in Chinese)
- **Croatia**
CROSSIM—Croatian Society for Simulation Modelling
- **Denmark**
DKSIM—Dansk Simuleringsforening (Danish Simulation Society)

- **Finland**
FinSim—Finnish Simulation Forum
- **France**
CNRS-GdR MACS—Groupe de Recherche “Modelisation, Analyse et Conduite des Systemes dynamiques” de CNRS
FRANCOSIM—Société Francophone de Simulation (Belgium, France)
SoFraSimS—Société Francophone de Simulation en Santé
- **Hong Kong**
SGA—Serious Games Association
- **Hungary**
HSS—Hungarian Simulation Society
- **India**
IAMMS—Indian Academy for Mathematical Modeling and Simulation
INDSAGA—Indian Simulation and Gaming Association
- **Italy**
ISCS—Italian Society for Computer Simulation
Liophant Simulation
MIMOS (Italian Movement for Modeling and Simulation)
SIMMED—SIMulation in MEDicine
Simulation Team
- **Japan**
JASAG—Japan Association of Simulation and Gaming
JSST—Japan Society for Simulation Technology
- **Korea**
KoSSH—Korea Society for Simulation in Healthcare
KSS—The Korea Society for Simulation (in Korean)
- **Kosova**
KA-SIM—Kosova Society for modeling and Simulation
- **Latvia**
LSS—Latvian Simulation Society
- **Malasia**
MASAGA—Malaysian Simulation & Gaming Conference
- **Netherlands**
DSSH—Dutch Society for Simulation in Health Care
SAGANET—Simulation and Gaming Association Derneği (Medical Simulation Association)
- **Poland**
PSCS—Polish Society for Computer Simulation (in Polish)
- **Puerto Rico**
ASICTEPROS—Asociación de Simulación Clínica y Tecnología Digital de Puerto Rico para Profesionales de la Salud
- **Romania**
ROMSIM—Romanian Society for Modelling and Simulation

- **Russia**
NSSM—National Society for Simulation Modeling in Russia
- **Singapore**
SGA—Serious Games Association
SSAGSg—Society of Simulation and Gaming of Singapore
- **Slovenia**
SLOSIM—Slovenian Society for Modelling and Simulation
- **Spain**
ASESP—Asociación Estudiantil de Simulaciones Parlamentarias
CEA-SMSG—Spanish Modeling and Simulation Group of the Spanish Society on Automation and Control
SESSEP—Sociedad Española de Simulación Clínica y Seguridad del Paciente
- **Taiwan**
TaiwanSG—Taiwan Simulation and Gaming Association
- **Thailand**
ThaiSim—The Thai Simulation and Gaming Association
- **Turkey**
BinSimDer—Bina Performansı Modelleme ve Simülasyonları Derneği
CASE—Center of Advanced Simulation and Education
Hemşirelikte Klinik Simülasyon Derneği
IBPSA TR—Bina performans Simülasyonları Komitesi
Medsimmer—Medikal Simülasyon Derneği
SSH—Sağlıkta Simülasyon Derneği
- **UK**
ASPiH—Association for Simulated Practice in healthcare
UKSIM—United kingdom Simulation Society (UK, Ireland)
- **USA**
AIMS—Advanced Initiative in Medical Simulation
TNA SEM CSL—Committee on Learning Science: Computer Games, Simulations, and Education; Center for Education, Board on Science Education; The National Academies
SIETAR-USA—Society for Intercultural Education Training and Research
SSH—Society for Simulation in Healthcare
URSGA—University of Rochester Simulation Gaming Association
WISER—Education and Simulation Improving Healthcare

Associations—by Region/Language

- **Americas**
CSSSA—Computational Social Science Society of the Americas (formerly NAACSOS)
- **Asia**
AFSG—Asian Federation for Serious Games
ASIASIM—Federation of Asian Simulation Societies

- **Asia-Pacific**
APSA—Asia-Pacific Simulation Association
- **Australia/New Zealand**
MSSANZ—Modelling and Simulation Society of Australia and New Zealand Inc.
- **Czech and Slovak Republics**
CSSS—Czech and Slovak Simulation Society
- **Dutch Benelux**
DBSS—Dutch Benelux Simulation Society
- **Europe**
BETA—Simulation of European Politics
EASA—ATOs—European Aviation safety Agency—Approved Training organizations
ECMS—European Council for Modeling and Simulation
ESSA—The European Social Simulation Association
ESSA SSSG—Special Interest Group on Social Simulation and Serious gGames
EUROSIM—Federation of European Simulation Societies
EUROSIS—The European Multidisciplinary Society for Modelling and Simulation Technology
SESAM—Society for Simulation in Europe
- **French**
FRANCOSIM—Societe de Simulation Francophone
- **German**
ASIM—German Simulation Society
- **Mediterranean and Latin America**
IMCS—International Mediterranean and Latin American Council of Simulation
- **North America**
NAACSOS—North American Association for Computational Social and Organizational Science (currently part of CSSSA)
NASAGA—North American Simulation and Gaming Association
- **Pacific Asia**
PAAA—Pacific Asian Association for Agent-based Approach in Social Systems Sciences
- **Scandinavia**
SIMS—Scandinavian Simulation Society
- **Swiss, Austrian, and German**
SAGSAGA—Swiss Austrian German Simulation and Gaming Association.

Research Centers/Groups

- ACIMS—Arizona Center for Integrative Modeling and Simulation
- AFSC—Asia Flight Simulation Centre
- AMCS MMSS—Australian Maritime College - Centre for Maritime Simulation

- AMSL—The Auburn Modeling and Simulation Laboratory
- BioSystems Group—(UCSF—University of California San Francisco)
- C&MSC—Cheshire and Merseyside Simulation Centre (Medical, Nursing, and Healthcare)
- CESAR—The Center for Exascale Simulation of Advanced Reactors
- CMS—Centre for Marine Simulation – Fisheries and Marine Institute of Memorial University of Newfoundland
- CMS-UNIPUNE—Centre for Modeling and Simulation, University of Pune, Pune, India
- Marseille, France
- CRESS—Centre for Research in Social Simulation
- CSBL—Centre for Simulation Based learning—McMaster University, Faculty of Health Sciences
- CSPS—Centre for Simulation and Patient Safety
- CWRU—Mt. Sinai Skills and Simulation Center
- FMSC—Fremantle Marine Simulation Centre
- IBM Cyberattack Simulation Test Centre
- ICSS—Institute for Complex Systems Simulation, University of Southampton, UK
- IST—School of Modeling Simulation Training (University of Central Florida)
- KMSC—Kohl's Mobile Simulation Center
- Liophant Simulation Club
- LSIS—Laboratoire des Sciences de l'Information et des Systèmes (Information and Systems Sciences Laboratory), Marseille, France
- SAG-ETH—Simulation and Animation Group (part of the Computer Graphics laboratory at the ETH in Zurich)
- MacEwanU—Clinical Simulation Centre
- MoSeS—Modelling and Simulation in e-Social Science
- MSC-LES—Modeling & Simulation Center—Laboratory of Enterprise Solutions (at Mechanical Department, University of Calabria)
- MSR—Israel Center for Medical Simulation
- MSUG—Michigan Simulation User Group
- NAIT—Centre for Advanced medical Simulation
- NATSEM—National Centre for Social and Economic Modelling, Australia
- NISAC—National Infrastructure Simulation and Analysis Center, USA
- RSRG—Reservoir Simulation Research Group, University of Calgary
- SAAB TCSC—Training and Simulation—Live Training
- SIRC—Simulation Innovation Resource Center (for Nurse Educators)
- SMS—Systems Modeling Simulation Laboratory at KAIST (Korea Advanced Institute of Science and Technology)
- SRG-LBNL—Simulation Research Group at Lawrence Berkeley National Laboratory (for building energy simulation software)
- SRG-ORMS—Simulation Research Group at Operational Research and Management Sciences Group at the University of Warwick
- SSC—Swedish Simulation Center

- UAH CMSA—University of Alabama in Huntsville—Center for Modeling, Simulation and Analysis
- UHBW Simulation Centre (Previously known as Bristol Medical Simulation Centre—BMSC)
- uOSSC—The University of Ottawa Skills and Simulation Centre
- VSIM—Centre for Advanced Studies in Visualization, Simulation & Modelling (Carleton University, Ottawa, ON, Canada).

Military Organizations

NATO

- M&S COE—NATO Modelling & Simulation Centres of Excellence
- NMSG—NATO Modelling and Simulation Group
- NMSG Team—**NMSG Specialist Team on Modelling and Simulation in NATO Federated Mission Networking**
- SAS—NATO Studies, Analysis and Simulation Panel.

Europe

ESM3—Systems of Systems, Space, Simulation & Experiment of EDA (European Defence Agency)

By Country

Australia

- ADC-WSC—Australian Defence College, Wargaming & Simulation Centre
- ADSTC—Australian Defence Simulation and Training Centre.

Canada

- AGDA T&SEC—Training and Engineering Centre, Kingston, Ontario
- Canadian Army Simulation Centre
- Simulation Canada.

Korea

- KBSC—Korean Battle Simulation Center.

Turkey

- MODSIMMER—Modeling and Simulation on Research and Development Center, USA
- DMSO—Defense Modelling and Simulation Office
- MOVES—Modeling, Virtual Environments & Simulation Institute. Naval Postgraduate School
- MSE—Defense Modelling and Simulation Enterprise
- MSIAC—Modelling and Simulation Information Analysis Center
- NDIA—M&SCom—Modelling and Simulation Committee of the Systems Engineering Division of NDIA (National Defence Industrial Association)
- SISO—Simulation Interoperability Standards Organization.
SISO-SCMs—Standing Study Groups (SSGs) of SISO

- CBMS SSG—Cloud-based Modelling & Simulation
- ENG TAM SSG—Exploration of Next Generation Technology Applications to Modeling and Simulation
- S&WG SSG—Simulation and Wargaming
- Simulation Australasia SSG
- SIM-SCALE SG—Simulation Scalability
- XR-INTEROP SG—XR Interoperability Standards.

Air Force

- AFAMS—Air Force Agency for Modeling and Simulation

Army

- AMSO—US Army Model and Simulation Office
- PEO STRI—US Army Program Executive Office for Simulation, Training, & Instrumentation.

Marine/Navy

- MCMSO—Marine Corps Modeling and Simulation Office
- MTWS—Marine Air Ground Task Force (MAGTF) Tactical Warfare Simulation
- NAVMSMO—Navy Modeling and Simulation Management Office
- NMSO—Navy Modeling and Simulation Office.

Space

- CMSO—Chief Modeling and Simulation Office—Department of the Air Force
- LSIST—Laboratory of Space Information and Simulation Technology.

A4—Additional Resources

Archives

I/ITSEC (The Interservice/Industry Training, Simulation and Education Conference)

Knowledge repository: <http://www.iitsecdocs.com/>

Search by keywords: <http://www.iitsecdocs.com/search>

Search by year: <http://www.iitsecdocs.com/volumes>

INFORMS Simulation Society North Carolina State University (NCSU) Simulation Archive

<https://connect.informs.org/simulation/simulation-resources/simulation-archive>

Journal of System Simulation—Archive from issue 1 (1989)

<https://www.oriprobe.com/journals/xtfzxb.html>

NC State University Libraries Computer Simulation Archive

<https://d.lib.ncsu.edu/computer-simulation/giving/>

Winter Simulation Conference Archive

Search by year: <https://informs-sim.org/>

Ethics

Ethics: The Society for Computer Simulation International (SCS) A Code of Professional Ethics for Simulationists <https://scs.org/ethics/>

Ethics: Society for Simulation in healthcare (SSH) Healthcare Simulationist Code Of Ethics

<https://www.ssih.org/Portals/48/SSH%20-Code-of-Ethics.pdf>

EU: SimE: Simulation Pedagogy in Learning Ethics in Practice of Health Care <https://simethics.eu/>

Sim ethics—What are our ethical obligations to future AI? <https://curio.io/stories/1gTFDzCe8tHrxOA0akLjZQ>

Lists of

List of Agent-Based Models (ABM) Researchers

<http://www.agent-based-models.com/blog/researchers/>

List of Simulationists (by Prof. François E. Cellier) <https://uweb.engr.arizona.edu/~cellier/people.html>

List of Systems Engineers (Wikipedia - https://en.wikipedia.org/wiki/List_of_systems_engineers) (WikiVisually—https://wikivisually.com/wiki/List_of_systems_engineers)

List of Systems Sciences organizations https://en.wikipedia.org/wiki/List_of_systems_sciences_organizations

List of Systems Scientists (Wikipedia—https://en.wikipedia.org/wiki/List_of_systems_scientists) (pictures—https://en.wikipedia.org/wiki/List_of_systems_scientists)

YouTube

https://www.youtube.com/results?search_query=simulation

Appendix B: Bios of the Contributors

Editors



Tuncer Ören Dr. Ören is a professor emeritus of computer science at the School of Electrical Engineering and Computer Science of the University of Ottawa, Canada. He has been involved with simulation since 1965.

Education: Dr. Ören's Ph.D. is in Systems Engineering from the University of Arizona, Tucson, AZ (1971). His basic education is from Galatasaray Lisesi, a high school founded in his native Istanbul, in 1481, and in Mechanical Engineering at the Technical University of Istanbul (1960).

His research interests include:

- advancing methodologies for modeling and simulation;
- agent-directed simulation;
- cognitive and emotive simulations (including representations of human personality, emotions, anger mechanisms, understanding, misunderstanding, and computational awareness);
- reliability, QA, failure avoidance;
- ethics;
- Body of Knowledge;
- terminology of modeling and simulation.

He authored and co-authored over 560 publications, including over 60 books and proceedings, and has contributed, since 1950s, to over 500 conferences and seminars held in 40 countries.

Distinctions: Dr. Ören is a fellow of SCS (2016) and an inductee to SCS Modeling and Simulation Hall of Fame (2011). He received **SCS McLeod Founder's Award** for Distinguished Service to the Profession (2017) and the Golden Award of Excellence **from the** International Institute for Advanced Studies in

Systems Research and Cybernetics (2018). He was selected by IBM Canada as one of the pioneers of computing in Canada (2005). A book was edited by Prof. Levent Yilmaz: *Concepts and Methodologies for Modeling and Simulation: A Tribute to Tuncer Ören*. Springer (2015).



Bernard P. Zeigler Dr. Zeigler is Chief Scientist for RTSync Corp and Professor Emeritus of Electrical and Computer Engineering at the University of Arizona, Tucson. Zeigler is a co-director of the Arizona Center for Integrative Modeling and Simulation. RTSync is a spinoff of ACIMS devoted to transferring DEVS-based technology to general use. He is also on the staff of the C4I Center of George Mason University.

Education: Dr. Zeigler's Ph.D. is in Computer/Communication Science from University of Michigan, Ann Arbor, MI (1968). He graduated with a B.S. (Eng. Phys.) from McGill University, Montreal (1962) and an M.S. (E.E.), Massachusetts Institute of Technology, 1964.

His research interests include:

- Methodology of Modeling and Simulation
- Software/Hardware support for M&S Development
- Modeling and Simulation of Healthcare Systems
- Model-based System Engineering
- Bridging the Gap between Cognitive Behavior and Neural Circuits

Distinctions: Zeigler is recognized as Computer Simulation Pioneer and internationally known for his 1976 foundational text *Theory of Modeling and Simulation*, revised for a third edition (Academic Press, 2018). He has published numerous books and research articles on the Discrete Event System Specification (DEVS) formalism. He is a fellow of the IEEE (for his invention of DEVS) and of the Society for Modeling and Simulation, International (SCS). He was inducted into the SCS Modeling and Simulation Hall of Fame in 2009. He received the highest awards offered by the SCS and INFORMS, the McLeod Founders Award, and the Lifetime Professional Achievement Award, respectively. Dr. Zeigler is speaking across the globe as IEEE Distinguished Visitor—2019–2021.



Andreas Tolk Andreas Tolk is Chief Scientist for Complex System Modeling in the Modeling and Analysis Innovation Center of The MITRE Corporation in Charlottesville, VA. He is also Adjunct Full Professor at Old Dominion University in Norfolk, VA. He holds a Ph.D. and M.Sc. in Computer Science from the University of the Federal Armed Forces of Germany.

His research interests include computational and epistemological foundations and constraints of model-based solutions in computational sciences and their application in support of model-based systems engineering, including the integration of simulation methods and tools into the systems engineering education and best practices. He is also contributing to the field of artificial societies for policy evaluation in the intersection of modeling and simulation, complex adaptive systems, and computational social sciences.

He published more than 250 peer-reviewed journal articles, book chapters, and conference papers and edited 14 textbooks and compendia on Systems Engineering and Modeling and Simulation topics. He is Fellow of the Society for Modeling and Simulation (SCS) and Senior Member of IEEE and the Association for Computing Machinery (ACM). He received multiple awards, including distinguished contribution awards from SCS and ACM and the inaugural technical merit award of the Simulation Interoperability Standards Organization (SISO).

Contributing Authors



Laurent Capocchi Laurent Capocchi was born in Bastia, Corsica, France. He received an M.Sc. in electrical engineering from “Ecole Supérieure d’Ingénieurs de Nice Sophia Antipolis” (ESINSA), Nice, France, in 2001, and the Ph.D. in Computer Science from University of Corsica “Pasquale Paoli”, Corte, France, in 2005. He is an Society for Modeling and Simulation International (SCS) member since 2001, and he is also a member of the Computer Science SISU Team of the “Sciences pour l’environnement” (SPE) Laboratory at the University of Corsica. His main research concerns the discrete event modeling and

simulation of complex systems by using the Discrete Event System Specification (DEVS) formalism.

His research field interest includes behavioral faults concurrent simulation, optimization via simulation, model-based system engineering, modeling and simulation as a service and reinforcement learning. He is a founder member of the DEVSImPy/DEVSImPy-mob suite open-source project, and he contributes actively to its development.



Lance Champagne Lance Champagne earned a B.S. in Biomedical Engineering from Tulane University, M.S. in Operations Research, and Ph.D. in Operations Research from the Air Force Institute of Technology (AFIT). He served as an officer in the US Air Force for 23 years and retired with the rank of Lieutenant Colonel. Currently, Lance is Assistant Professor of Operations Research in the Department of Operational Sciences at the Air Force Institute of Technology with research interests including simulation of autonomous system behavior, agent-based combat simulation, multivariate analysis techniques, and image/video classification. His email address is lance.champagne@afit.edu.

neural network design for



Su Chunhui Su Chunhui, master's degree, graduated from Beijing Institute of Technology, majored in Computer Science and Technology. As a vice department manager in the digital engineering department of China Aerospace Science and Industry Corporation Limited, CASICloud-Tech Co., Ltd., Beijing Aerospace Smart Manufacturing Technology Development Co., Ltd., she has long been engaged in the technical research of industrial internet platform, production simulation, digital twin, and VR/AR. She has participated in the research, development of the INDICS Platform (Casicloud), focused on the virtual production

subsystem. More than 10 papers were published in academic journals and conferences.



Esteban Clua Esteban Clua is a professor at Universidade Federal Fluminense and coordinator of UFF Medialab, CNPq researcher 1D, Scientist of the State of Rio prize in 2019 and Young Scientist of the State of Rio in 2009 and 2013. He is undergraduate in Computer Science by Universidade de São Paulo and has master and doctor degrees by PUC-Rio. His main research and development area are Digital Games, Virtual Reality, GPUs and Visualization.

He is one of the founders of SBGames (Brazilian Symposium of Games and Digital Entertainment) and was the president of Game Committee of the Brazilian Computer Society from 2010 through 2014. Today he is the commission member for Brazil at the Technical Committee of Entertainment at the International

Federation of Information Processing (IFIP) and honorable member of the board council of ABRAGAMES (Brazilian Association for Game Development). In 2015 he was nominated as NVidia CUDA Fellow. In 2007 he received the prize from ABRAGAMES as the main contributor from the academia for the development of the game industry at Brazil. Esteban is member of the program committee of most digital entertainment conferences. He is today the coordinator of the NVIDIA Center of Excellence that is located at the CS Institute of UFF. Esteban belongs to the council of innovation of the Culture Secretary of the State of Rio, is member of the permanent commission of Rio Criativo, Member of the permanent board of innovation and technology at the Legislative board of Rio and member of the Innovation Agency at UFF.



Nicolas Daclin Nicolas Daclin is Associate Professor at IMT-Mines Alès, Laboratoire des Sciences des Risques (LSR). He earned his Ph.D. from University Bordeaux 1 in 2007 in interoperability of organizations and his HdR from University of Montpellier in 2017 in evaluation of non-functional requirements in collaborative organization.

Research topics are systems engineering enterprise modeling, systems interoperability, requirements engineering, process modeling and automation applied in the fields of health, crisis management and industrial systems He has been involved in international projects (INTEROP Noe, ATHENA IP, REHSTRAIN) national projects (ISYCRI, COR-EGI, CARIONER I/II, RESIIST).



Andrea D'Ambrogio Andrea D'Ambrogio is associate professor at the Department of Enterprise Engineering of the University of Roma Tor Vergata, where he leads the Software Engineering Laboratory (sel.uniroma2.it). He has formerly been research associate at the Concurrent Engineering Research Center of the West Virginia University and director of the postgraduate master's degree program in Systems Engineering at the University of Roma Tor Vergata. His research interests are in the areas of model-driven systems and software engineering, systems dependability engineering, business process management, and distributed and

web-based simulation. In such areas, he has participated in several projects at both European and overseas level and has authored more than 150 journal/conference papers.

He has served as a general chair and/or program chair of various international events, such as the Spring Simulation Conference, the Summer Simulation conference, the Theory of Modeling and Simulation (TMS) Symposium, the DS-RT conference, the European Simulation and Modelling Conference, and the

IEEE WETICE conference. In 2010, he started the IEEE International Workshop on Collaborative Modeling and Simulation (CoMetS) and in 2011 the SCS International Workshop on Model-Driven Approaches for Simulation Engineering (Mod4Sim). He does scientific advisory work for industries and for national and international organizations, and he is a member of IEEE, IEEE Computer Society, ACM, SCS, and INCOSE. He is the president-elect and members of the Board of Directors of the Society for Modeling and Simulation International (SCS).



Paul K. Davis Paul K. Davis is Principal Researcher at RAND (formally retired but active as an adjunct) and as Professor of Policy Analysis in the Pardee RAND Graduate School. His research has been in strategic planning, strategy, resource allocation, decision-making theory, deterrence and influence, modeling and simulation for analysis (e.g., game-structured simulation, multi-resolution modeling, exploratory analysis, and applying social science to policy analysis).

His most recent book is Paul K. Davis, Angela O'Mahony, and Jonathan Pfautz (eds.), *Social-Behavioral Modeling for Complex Systems* (Wiley, 2019). Dr. Davis studied chemistry at the University of Michigan and received his Ph.D. in theoretical chemical physics from the Massachusetts Institute of Technology. He has worked at the Institute for Defense Analyses, the US State Department, the US Department of Defense as Senior Executive in Program Analysis and Evaluation, and—for many years—at the RAND Corporation. Many of his publications are available from his RAND *website* (https://www.rand.org/about/people/d/davis_paul.html).



Breno Bernard Nicolau de França Breno is a professor at the Computing Institute (UNICAMP) and a coordinator of the LASER research laboratory (Laboratory for Software Engineering and Reliability), conducting research on Empirical Software Engineering, Lean and Agile Software Development, Software Architecture, and Computer Simulation.

He got his Ph.D. in Systems Engineering and Computer Science at COPPE/UFRJ, where Breno also concluded a postdoctoral fellowship. He got his master's and bachelor's degrees in Computer Science at Universidade Federal do Pará (UFPA). Over the years, Breno had several collaborations with public and private organizations in the context of research and development for technology acquisition, evaluation, and transfer supported by experimental methods, software process improvement, and software engineering education and training.



Saikou Y. Diallo Saikou Y. Diallo is Chief Scientist, Research Associate Professor, Virginia Modeling, Analysis and Simulation Center (VMASC), Old Dominion University. He is a scientist and educator in the domain of innovation and equal access to technology. He is passionate about using modeling, simulation, and analytics to study the connection between people on all spectrums and artificial beings.

Dr. Diallo works with a multi-disciplinary team of artists, scientists, humanities scholars, and engineers to tackle societal issues that transcend disciplines. His current work involves modeling religion, culture, and civilizations and the use of artificial intelligence to improve independent living. He is also involved in developing platforms and solutions to promote an inclusive use of technology across all spectrums.

Dr. Diallo graduated with a M.S. in Engineering in 2006 and a Ph.D. in Modeling and Simulation in 2010 both from Old Dominion University. He is current President of the Society for Modeling and Simulation International (SCS) and serves on several boards in the United States. Dr. Diallo has over one hundred publications in peer-reviewed conferences, journals, and books.



Rodrigo Pereira dos Santos Dr. Santos is a professor of software engineering and information systems at the Department of Applied Informatics from the Federal University of the State of Rio de Janeiro (UNIRIO). He has worked with systems modeling and educational games since 2007 and is a co-editor of the Grand Research Challenges on Games and Entertainment Computing in Brazil (GrandGamesBR 2020–2030). Dr. Santos holds doctorate (2016) and master's (2010) degrees in Computer Science and Systems Engineering from the Federal University of Rio de Janeiro (COPPE/UFRJ).

His bachelor's degree in Computer Science was received from the Federal University of Lavras (2007). He was Academic Visitor at University College London and Postdoctoral Researcher at COPPE/UFRJ. He is Head of the Complex Systems Engineering Laboratory (LabESC), and his research interests include complex systems engineering (especially software ecosystems and systems-of-systems) and computer science education and training. He has published more than 200 studies and has organized more than 20 workshops and symposia. Dr. Santos has been a member of the Education Evaluating Commission at the Brazilian Ministry of Education since 2011. As a Brazilian Computer Society (SBC) member, he is Coordinator of the Special Commission of Information Systems, Member of the Special Commission of Games and Entertainment Computing, Member of the Special Commission of Software Engineering, and Member

of the Interest Group of Computer Science Education. He was Editor-in-Chief of *iSys: Brazilian Journal of Information Systems* (2017–2021). He also worked in several projects related to software process improvement and reuse management in large companies at Coppetec Foundation (2008–2017). Dr. Santos has been an IEEE and ACM member.



Guy Doumeings Guy Doumeings is Emeritus Professor at Bordeaux University in which he has developed his career since 1968 after an initial period as Engineer in Aerospace industry and as well General Manager of INTEROP-VLab, the virtual laboratory in Enterprise Interoperability since 2007. He is a specialist in the domain of Enterprise Modeling (EM), Business Process Management (BPM), Enterprise Interoperability (EI), System Performance Evaluation, and Implementation of Industry 4.0 solutions.

He is the main author of the GRAI Model and the GRAI Integrated Methodology (GIM) since the 1980s. Guy has managed more than twenty important European and International projects during the last 30 years in his competence domains (PIRELLI, BRITISH AEROSPACE, IBERIA, SIEMENS, SCANIA, AEROSPATIALE, ALCATEL, SNECMA, SCHLUMBERGER, LYONNAISE DES EAUX, INTEROP-NoE but also with several SMEs ...). He has published more than 300 articles and four books. He is invited regularly to deliver conference in International Congress and to manage professional seminars. He acts also as International Consultant.



Umut Durak Umut Durak is Group Leader for Assured Embedded Systems at the German Aerospace Center (DLR) and Professor for Aeronautical Informatics at the Clausthal University of Technology. His research interests concentrate on simulation and model-based approaches in engineering of airborne software-intensive systems. He has published 5 books and more than 80 papers in various conference proceedings and journals. He served as Vice President at the Society for Computer Simulation International (SCS) between 2019 and 2021. He is currently Speaker of the Arbeitsgemeinschaft Simulation (ASIM) Section

—Basic Principles and Methods in Simulation (GMMS)—and an associate fellow of the American Institute of Aeronautics and Astronautics (AIAA).



Juan M. Durán Juan M. Durán is an assistant professor at the Delft University of Technology. He previously studied computer science and philosophy at the National University of Córdoba (Argentina). During this time, he worked at the National Space Activities Commission (Argentina) on several projects, including software engineering and programming neural networks, parallel processing, and remote sensing. He obtained his Ph.D. at the Cluster of Excellence SimTech—University of Stuttgart (Germany). Before joining TU Delft, he was a research associate at the High-Performance Computing Centre Stuttgart,

University of Stuttgart. In 2019, he won the *Herbert A. Simon Award* for outstanding research in computing and philosophy. This award is offered by the International Association for Computing and Philosophy (IACAP) and recognizes scholars at an early stage of their academic career who are likely to reshape debates at the nexus of computing and philosophy through their original research. He received this award for his work on computer simulations: “the board well recognizes the significant early contributions you have made staking out new ground in the epistemic and normative dimensions of computer simulations. We eagerly anticipate the impact your scholarship will have on many areas beneath the umbrella of computing and philosophy and wish to acknowledge your continued success with this award”. His research interests focus on the intersection between the philosophy of science and technology with ethics and values. He has extensively published on issues regarding computer simulations, big data, scientific models, and laboratory experimentation. His current research also includes themes from artificial intelligence and the philosophy of medicine.



Wenhui Fan Wenhui Fan is a professor at Tsinghua University. He received the B.S. degree in 1990 from the Department of Materials Science and Engineering, Northwest University of Technology, China. He received the M.S. degree in 1995 from the Department of Materials Science and Engineering, Harbin Institute of Technology, China. He received the Ph.D. degree in 1998 from the Department of Mechanical Science and Engineering, Zhejiang University, China. He served as the vice president of China Simulation Federation (CSF). He serves as Associate Editor-in-Chief of Complex System Modeling and Simulation and Associate Editor-in-Chief of Journal of System Simulation. He authored and co-authored more than 120 papers. He authored and co-authored more than 6 books including Introduction to System Simulation (Tsinghua University Press, 2010), Modeling and Simulation of Continuous System (Electronic Industry Press, 2010), Modeling and Simulation of Discrete Event System (Electronic Industry Press, 2011), etc. His

He authored and co-authored more than 6 books including Introduction to System Simulation (Tsinghua University Press, 2010), Modeling and Simulation of Continuous System (Electronic Industry Press, 2010), Modeling and Simulation of Discrete Event System (Electronic Industry Press, 2011), etc. His

research interests include complex system modeling and simulation, multi-agent simulation, multi-agent reinforcement learning (MARL), M&S for manufacturing and medical systems, etc.



Paul Fishwick Paul Fishwick is Distinguished University Chair of Arts and Technology (ATEC) and Professor of Computer Science. He has six years of industry experience as a systems analyst working at Newport News Shipbuilding and at NASA Langley Research Center in Virginia. He was on the faculty at the University of Florida from 1986 to 2012 and was Director of the Digital Arts and Sciences Programs. His Ph.D. was in Computer and Information Science from the University of Pennsylvania. Fishwick is very active in modeling and simulation, as well as in the bridge areas spanning art, science, and engineering. He pioneered the area of esthetic computing, resulting in an MIT Press edited volume in 2006.

He is Fellow of the Society for Computer Simulation, served as General Chair of the Winter Simulation Conference (WSC), was WSC Titan Speaker in 2009, and has delivered over 25 keynote addresses at international conferences. He was Chair of the Association for Computing Machinery (ACM) Special Interest Group in Simulation (SIGSIM) four years from 2012 to 2016. Fishwick has over 240 technical publications and has served on all major archival journal editorial boards related to modeling and simulation, including ACM Transactions on Modeling and Simulation (TOMACS) where he was a founding area editor of modeling methodology in 1990. He is CEO of Metaphorz, LLC which assists the State of Florida in its catastrophe modeling software engineering auditing process for risk-based simulation for hurricanes and floods.



Hendrik Folkerts Hendrik Folkerts is a Ph.D. student at the University of Applied Sciences in Wismar, Germany. He holds a bachelor's and a master's degree in Electrical Engineering and Information Technology. During his master's thesis, he developed a prototype software supporting the specification of System Entity Structures (SES). For his Ph.D. thesis, he developed a prototype software supporting the specification of models using the SES and a model base (MB) according to the SES/MB approach and its extensions for automatic model generation, execution, and evaluation. In particular, he examines possibilities to generate models for multiple simulators using the SES/MB approach.



Nico Formanek Nico Formanek studied Physics and Philosophy at TU Darmstadt. Currently, he is the deputy leader at the Department of Philosophy of Computational Methods, High-Performance Computing Center, University of Stuttgart. He has published work on the philosophy of computer simulations, including issues on epistemology, ethics, and their societal value.



Rhys Goldstein Rhys Goldstein is Sr. Principal Research Scientist at Autodesk Research. He has a master's degree in biomedical engineering from Carleton University (2009) and a bachelor's degree in engineering physics from the University of British Columbia (2003).

Rhys's contributions to modeling and simulation include award-winning papers on the design of visual modeling interfaces, the representation of time in discrete event systems, and the simulation of human behavior in buildings. He has served on the organizing committees of the 2014 and 2015 Simulation for Architecture and Urban Design (SimAUD) Symposium and the 2017 Theory of Modeling and Simulation (TMS/DEVS) Symposium. He has also served as an associate editor of *Simulation*. Rhys is the lead developer of the SyDEVS open source C++ library for systems modeling, as well as the VASA Dynamo package for voxel-based architectural space analysis.



Cláudio Gomes Cláudio Gomes is an Assistant Professor at Department of Engineering at Aarhus University. His research is centered on co-simulation and for good reason. During his Ph.D., he worked full time in mapping the state of the art on co-simulation, only to discover that there was an extensive body of knowledge on the topic, but that it was scattered across different disciplines. He then worked toward unifying the terminology used, identifying remaining challenges, then addressing some of those. For instance, he has connected the analysis of adaptive co-simulation algorithms to the joint spectral radius theory, leading to contributions not just in the former field, but also in the later field.

Many challenges remain to be solved, and more will be unfolded, as systematic approaches to producing co-simulations mature. Cláudio has helped the community by being in the program committees of Springsim and Summersim conferences and the CoSimCPS workshop.



Hezam Haidar Hezam Haidar is currently working for INTEROP-VLab as a research and innovation engineer. He earned his Ph.D. (University of Grenoble) in industrial engineering in open innovation mechanisms for the improvement of university-SME collaboration. Before that, he received two master's degrees (University of Grenoble): one in Industrial Process Automation and second master's in Business Development and Administration. He obtained an engineering degree in electronics majoring in telecommunication from Multimedia University.

He was previously Automation Engineer in Siemens power plant, Yemen. Since December 2019, he works for INTEROP-VLab with the core activity in enterprise modeling and interoperability also in business development. He is active in multiple European projects such as Smart4Health, DIH4CPS, and i4Q. Research interests: Business process modeling, enterprise modeling and interoperability, industrial and health systems, academia-industry collaboration, and open innovation.



Maâmar El Amine Hamri Maâmar El Amine Hamri has obtained his Ph.D. in computer science in 2005 from Aix-Marseille III University. In 2007, he joined the same university as maître de conférences for teaching in computer science department and conducting research in LSIS laboratory in the field of modeling and simulation of discrete event systems and especially in DEVS. In 2017, he obtained his Habilitation à Diriger des Recherches from Aix-Marseille Université. After a long research carrier in the field of DEVS M&S, he published about fifty articles in international journals and conferences well known by the M&S community. He also was a member of many program committees and reviewed many articles.

Dr. Hamri supervised 4 Ph.D. students all in the field of DEVS M&S as attested by the numerous articles co-authored. He participated also at many scientific projects implying industrial partners like ST-Microelectronics of Rousset world leader in designing smart and memory cards. He developed many simulations for different applications like circuits, networks on chip, forest fire, etc. He also participated and continues to participate actively to promote the techniques of M&S in different training courses for students.

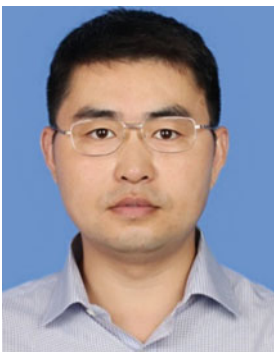


Alison Harper Alison Harper is a research fellow in Operations Research and Data Science applied to healthcare processes at University of Exeter Medical School, in the Peninsular Collaboration for Health Operational Research Development. She obtained her Ph.D. at the University of Exeter Business School in real-time simulation and short-term decision support. Her research interests are applied health and social care modeling, simulation, and data analytics to improve understanding and inform improvement for healthcare processes.



Raymond R. Hill Raymond R. Hill earned a B.S. in Mathematics from Eastern Connecticut State University, M.S. in Operations Research from the Air Force Institute of Technology (AFIT), and a Ph.D. from The Ohio State University. He retired from the US Air Force after 23 years at the rank of Lieutenant Colonel. Currently, Ray is Professor of Operations Research in the Department of Operational Sciences at the Air Force Institute of Technology with research interests including applied simulation, applied statistical analysis and experimental design, and heuristic optimization.

He is Editor-in-Chief for the Military Operations Research journal, Co-Editor-in-Chief of the Journal of Defense Analytics and Logistics, Associate Editor for Quality Engineering as well as the Journal of Simulation. His email address is rayrhill@gmail.com and rhill@afit.edu.



Baocun Hou Hou Baocun, Ph.D., a researcher, a chairman of China Aerospace Science and Industry Corporation Ltd. CASICloud-Tech Co., Ltd., Beijing Aerospace Smart Manufacturing Technology Development Co., Ltd., the deputy chief architect in the Cloud Manufacturing Field of China Aerospace Science and Industry Corporation Ltd., the deputy leader of the experimental platform group of Alliance of Industrial Internet, the deputy leader of the platform AD Hoc Group, and the leader of production-study-research Cooperation and Application Group in AIIA of China. He has long been engaged in the technical research of

smart cloud manufacturing/smart manufacturing, industrial internet platform, modeling and simulation of complex systems, and software system engineering.

He has participated in the research, development, and management of China's first and the world's first industrial Internet platform—the INDICS Platform (Casicloud). His related research results won 2s-level awards of National defense

science and technology progress. More than 60 papers were published in academic journals and conferences in domestic and abroad. He also co-published 2 academic work.



Umang Kant Umang Kant is currently pursuing Ph.D. from Delhi Technological University, Delhi, India, at the Department of Computer Science and Engineering as a full-time research scholar. She has done her M. Tech. from School of ICT, Gautam Buddha University, Greater Noida, Uttar Pradesh, India, in Intelligent Systems and Robotics and B.E. in Computer Science from Xavier Institute of Engineering, Mumbai, Maharashtra, India. Her specialization is in Artificial Intelligence, Machine Learning, Internet of Things, and Predictive Computing, among others.

She has worked as Assistant Professor in the department of Computer Science and Engineering at the Krishna Engineering College, Ghaziabad, Uttar Pradesh, India, and prior to joining Krishna Engineering College, she worked as Assistant Professor at U.I.E.T, Kurukshetra University, Kurukshetra, Haryana, India, and as Lecturer at Xavier Institute of Engineering, Mumbai, Maharashtra, India. She has guided several students in their B.Tech. projects and one student in M.Tech. thesis. She has also published around 10 technical research papers, 5 book chapters apart from other ongoing research work and has 2 patents to her credit.



Azam Khan Azam Khan is Co-founder and CEO of Trax.Co., developing tools for encoding codes, regulations, and standards for the built environment. He founded the Symposium on Simulation for Architecture and Urban Design (SimAUD) in 2010, has served as Adjunct Professor of Computer Science at the University of Toronto, and has been Velux Guest Professor at The Royal Danish Academy of Fine Arts, School of Architecture, Design and Conservation. He was also a member of the Technical Advisory Committee of CIFE at Stanford University, a research center for Virtual Design and Construction. Azam has been Director of

Complex Systems Research at Autodesk and previously was Research Scientist at Alias Systems. He has co-authored over 80 articles in modeling and simulation theory and practice, visual analytics, visual cognition, human Bayesian inference, human-computer interaction, sustainability, sensor networks, and architectural design. Azam received his B.Sc. and M.Sc. in Computer Science at the University of Toronto and his Ph.D. in Computer Science at the University of Copenhagen.



Yuanjun Laili Yuanjun Laili is currently a lecturer (Assistant Professor) at School of Automation Science and Electrical Engineering, Beihang University (Beijing University of Aeronautics and Astronautics), China, since September, 2015. She obtained her B.S., M.S., and Ph.D. degrees in 2009, 2012, and 2015 from School of Automation Science and Electrical Engineering at Beihang University, China. From 2017 to 2018, she worked as a research scholar at the Department of Mechanical Engineering, University of Birmingham, UK. She was elected in the “Young Talent Lift” project of China Association for Science and

Technology and was awarded as the “Young Simulation Scientist” of the Society for Modeling & Simulation International (SCS). She is an associate editor of “International Journal of Modeling, Simulation and Scientific Computing”. Her research interests include system modeling and simulation, evolutionary computation, and optimization in manufacturing systems. She is the author of one monograph and 28 journal and conference articles.



Bo Hu Li Professor Li, Bo Hu is Academician of Chinese Academy of Engineering, Ph.D. Adviser, Honorary President in the School of Automatic Science and Electrical Engineering at Beijing University of Aeronautics and Astronautics (BUAA), Honorary President of China Simulation Federation (CSF), Honorary Co-Chief-Editor of the journal “International Journal of Modeling, Simulation, and Scientific Computing”, and Honorary Director of China National Intelligent Manufacturing.

He was Director of Beijing Institute of Computer Application & Simulation Technology and Beijing Simulation Center, as well as President of the School of Automatic Science and Electrical Engineering in BUAA. He was the president of Chinese Simulation Federation, the first president of Federation of Asian Simulation Societies (ASIA-SIM), and a member of the council of directors of Society for Modeling & Simulation International (SCS). In addition, he was the director of Expert Committee of Contemporary Integrated Manufacturing System (CIMS) Subject in Chinese National High Technology Research and Development Plan.

In the fields of simulation and intelligent manufacturing, he authored or co-authored 400 papers, 15 books, and 4 translated books. He received 1 first-class scientific award and 3s-class scientific awards from China State and 17 scientific awards from the Chinese Ministries. In 2012, he received the Lifetime Achievement Award from the SCS and the “National Excellent Scientific and Technical Worker” honorary title from the Chinese Association for Science and Technology. In 2017,

he received the Lifetime Achievement Award from the China Computer Federation (CCF). In 2019, he received the Lifetime Achievement Award from the CSF.



Ting Yu Lin Ting Yu Lin obtained his Ph.D. in Control Science and Engineering, 2013, from the School of Automation Science and Electrical Engineering and the Honors College (an elite program) of Beihang University (BUAA), Beijing, China. He received his B. Eng. in Automatic and Information Technology, 2007, from the School of Automation Science and Electrical Engineering of Beihang University (BUAA), Beijing, China.

Since 2014, he works for the Beijing Simulation Center. He is now a senior engineer in State Key Laboratory of Intelligent Manufacturing System Technology which is undertaken by Beijing Simulation Center. His research interest includes complex system modeling and simulation and cloud simulation.

He authored and co-authored 70 papers, 10 books and chapters in the fields of his major and has got a first-class Scientific and Technological Progress Award of from a Ministry, China, in 2018. He is now a member of China Simulation Federation (CSF), and the convenor of the Study Group 7 (SG7) of ISO/TC 184/SC 5 for Interoperability of Simulation Models on Different Platforms.



Yang Liu Yang Liu, a senior engineer, received the B. S. degree in measurement and control technology and instruments from Xi'an University of technology in 2007 and M.S. degree in machinery and electronics engineering from Beijing University of technology in 2010.

From 2010 to 2016, she was Process Designer with Beijing Aerospace Xin feng Machinery Co., Ltd. and engaged in complex product process design. Since 2016, she was Technical Director with Beijing Aerospace Manufacturing Technology Co., Ltd. Her research interest includes fundamental study of intelligent manufacturing, industrial internet, artificial intelligence, and machine vision.

Author's awards and honors include publishing 10 papers and applying for 3 invention patents, co-published 2 academic work, and participating in research result "intelligent manufacturing cloud platform led by the new generation of artificial intelligence technology" was listed as one of the top 10 international intelligent manufacturing technological advances in 2019.



Margaret L. Loper Margaret L. Loper is Associate Director for Operations and Chief Scientist for the Information and Communications Laboratory at the Georgia Tech Research Institute. Margaret has worked in modeling and simulation (M&S) for more than 30 years, specifically focused on parallel and distributed systems. She is a founding member of the Simulation Interoperability Standards Organization (SISO) and received service awards for her work with the Distributed Interactive Simulation (DIS) and High-Level Architecture (HLA) standards and the DIS/SISO transition. Her research contributions are in the areas of

temporal synchronization, simulation testing, and simulation communication protocols.

Margaret has taught simulation courses for both academic and professional education for more than 15 years. She is a founding member of Georgia Tech's Professional Masters in Applied Systems Engineering degree program, where she teaches the core M&S course. Margaret started the M&S professional education certificate through the Georgia Tech School of Professional Education and has taught hundreds of students since it started in 2007. In 2015, she published a book on modeling and simulation in the systems engineering life cycle.

Her current research is focused on computational trust algorithms, edge intelligence, and situational awareness in smart cities and megacities.



Saurabh Mittal Dr. Saurabh Mittal is currently Principal Scientist and Project Lead supporting US Air Force Programs at the MITRE Corporation. Previously at MITRE, he has held various other roles such as Chief Scientist for Simulation, Experimentation and Gaming Department, M&S Subject Matter Expert, Technical Advisor for various government sponsors, and Principal Investigator for multiple internal MITRE research efforts. He received Ph.D. (2007) in Electrical and Computer Engineering (ECE) with dual minors in Systems and Industrial Engineering and Management and Information Systems, M.S. (2003) in ECE from the

University of Arizona, Tucson. and B.Tech. (2001) in Electrical Engineering from Jamia, Millia Islamia, India.

Previously, he contributed to modeling and simulation (M&S) efforts at National Renewable Energy Lab, Department of Energy, Colorado; Link Simulation and Training at US Air Force Research Lab at Wright Patterson Air Force Base, Dayton, Ohio; Northrop Grumman Information Technology at DISA, Ft. Huachuca, Arizona, and at the University of Arizona, Tucson.

He has co-authored over 100 publications as book chapters, journal articles, and conference proceedings including 5 books, covering topics in the areas of complex systems, system of systems, complex adaptive systems, emergent behavior, modeling and simulation (M&S), and M&S-based systems engineering. He served as President for the Society for Modeling and Simulation International (SCS) for FY21, President-Elect (FY20) and Member of the Board of Directors (FY18–20). He has also served as General Chair, Vice General Chair, Program Chair, and Track Chair during the 2013–2021 period in Spring, Summer, and Winter Simulation Conferences. He serves on the editorial boards of Transactions of SCS and Journal of Defense M&S. He is a recipient of the US DoD's highest civilian contractor recognition: Golden Eagle award and Outstanding Service and Professional Contribution awards by SCS.



Nav Mustafee Prof. Nav Mustafee is Professor of Analytics and Operations Management and Deputy Director for the Centre for Simulation, Analytics and Modelling (CSAM) at the University of Exeter Business School, UK. His research interests are in hybrid modeling and simulation, data-driven analytics, and real-time simulation for short-term decision-making. Another strand of his work is on Bibliometrics and meta-data analysis of research domains. In terms of application area, his focus is on healthcare. He is Founder and Co-chair of the Health and Care IMPACT Network, a collaboration between University of Exeter and several NHS Trusts.

Through this network, Nav led the research, development, rollout, and evidence synthesis of the NHSquicker digital platform. The platform provides real-time data on wait times from ~25 ED departments and minor injury units in the South West of England and nudges users to the most appropriate centers of urgent care. The objective is to shape demand for urgent services and to reduce overcrowding at EDs.

Nav publishes in European Journal of Operations Research, Computers and Operations Research, Journal of the Operational Research Society, ACM TOMACS, and Scientometrics. He is Associate Editor for Simulation, Health Systems, and the Journal of Simulation (JOS). He has guest-edited several special issues in journals such as TOMACS, Simulation, and JOS.

Nav has held leadership positions at several international conferences, including as General Chair (SpringSim), Program Chair/Co-Chair (ACM SIGSIM PADS, UK OR Society Annual Conference), and Proceedings Editor (Winter Simulation Conference). His email is n.mustafee@exeter.ac.uk. His webpage is <http://sites.google.com/site/navonilmustafee/>.



Valdemar Vicente Graciano Neto Dr. Graciano Neto is a professor of software engineering and information systems at the Informatics Institute of the Federal University of Goiás, in Goiânia, Brazil. He has worked with simulation since 2016. Dr. Graciano Neto is a D. Sc. and has a double degree on Computer Science and Computational Mathematics from the University of São Paulo, São Carlos, Brazil, and Informatics from IRISA Labs in University of South Brittany, Vannes, France (2018). His bachelors' and master's degrees on Computer Sciences were received from Federal University of Goiás, Brazil (2009 and 2012, respectively).

His research interests include (1) modeling and simulation for software engineering, (2) software architecture, (3) model-driven engineering, (4) information systems, (5) systems-of-systems (SoS) and systems-of-information systems (SoIS), and (6) smart cities. He has published more than 70 studies over the past years, including papers in proceedings of international conferences, important journals, and book chapters. He has also organized workshops and symposia. Dr. Graciano Neto was the coordinator of the Special Committee on Information Systems (CESI) of the Brazilian Computer Society (SBC) during 2018–2019 period, being also an elected member during 2015–2017. He was the general chair of the Brazilian Symposium on Information Systems (SBSI) in 2015, and he has organized the workshop series on modeling and simulation for software-intensive systems (MSSiS). Dr. Graciano Neto has also been a member of scientific communities, such as SCS (2019), SBC (2009–2020), and ACM.



Mohammad S. Obaidat Professor Mohammad S. Obaidat is an internationally known academic/researcher/scientist/scholar. He received his Ph.D. degree in Computer Engineering from Ohio State University, USA. He has received extensive research funding and published to date (2021) over one thousand and two hundred (1200) refereed technical articles—about half of them are journal articles, over 100 books, and over 75 book chapters. He is the editor-in-chief of 3 scholarly journals and the editor of many other international journals. He is the founding editor-in-chief of Wiley Security and Privacy Journal. Moreover, he is the founder/co-founder of 5 International Conferences.

Among his previous positions are Advisor to the President of Philadelphia University for Research, Development, and IT, President and Chair of Board of Directors of Society for Modeling and Simulation International (SCS), Dean of College of Engineering at Prince Sultan University, Chair and Tenured Professor at the Computer and Information Science Department at Fordham university, USA,

Chair and Tenured Professor of Computer Science Department and Director of Graduate Program at Monmouth University, USA, Tenured Professor at King Abdullah II School of Information Technology, University of Jordan, PR of China Ministry of Education, Distinguished Overseas Professor at University of Science and Technology, Beijing, China, and Honorary Distinguished Professor at Amity University. He is now Founding Dean and Professor, College of Computing and Informatics at University of Sharjah, UAE. He has received numerous worldwide technical and service awards, has chaired over 175 international conferences, and given over 175 worldwide keynote speeches.

He is Life Fellow of IEEE and Fellow of SCS.



Ernest H. Page Ernest H. Page is DARPA Portfolio Manager at The MITRE Corporation. Previously, he served as Chief Engineer within the Modeling, Simulation, Experimentation and Analytics Technical Center and Founding Director of MITRE's Simulation Experimentation and Analytics Lab (SEAL). He holds a Ph.D. in Computer Science from Virginia Tech. With a research interest in simulation modeling methodology and distributed systems, he has served as Principal Investigator on numerous government-funded and Independent Research and Development (IR&D) projects. He has held a variety of senior advisory roles,

including: Technical Adviser for the US Army Model and Simulation Office, Chief Scientist for the US Army Future Combat Systems (FCS) Modeling Architecture for Research and Experimentation (MATREX), and Member of the Defense Science Board Task Force on Gaming, Exercising and Modeling and Simulation. Dr. Page has published over 50 peer-reviewed articles in the areas of simulation modeling methodology and parallel and distributed systems. He served as Chair of the Association for Computing Machinery (ACM) Special Interest Group on Simulation (SIGSIM), the Board of Directors of the Winter Simulation Conference (WSC), and serves on the editorial boards of Transactions of the Society for Modeling and Simulation International, Journal of Defense Modeling and Simulation, and Journal of Simulation.



Thorsten Pawletta Thorsten Pawletta is Full Professor in Applied Computer Science at the University of Applied Sciences in Wismar, Germany, since 1994. He has M.Eng. and Ph.D. (Dr.-Ing.) equivalent degrees in mechanical engineering. The focus of his thesis was the application of modeling and simulation (M&S) methods in industrial automation. After his Ph.D. and working for the industry, he spent four years at the Department of Computer Science at the University of Rostock in Germany.

His long-term research interests include the theory of M&S, the combination of M&S methods with other computational methods, and the software engineering conversion of theoretical approaches into tools for engineers.

He is particularly interested in the further development of the DEVS theory and the concept of SES/MB. He has published several book contributions, journal articles, and a number of peer-reviewed conference papers. He is a long-standing member of SCS and the European Simulation Community (Eurosim/ASIM). He is also on the board of Eurosim/ASIM and the co-editor of the SNE—Simulation Notes Europe journal of Eurosim/ASIM.



Balqies Sadoun Balqies Sadoun received her M.S. in Civil Engineering from the Ecole Nationale des Travaux Publics de L'Etat, Lyon, France. She received another M.S. and Ph.D. degrees from The Ohio State University, Columbus, Ohio, USA, where she was on a scholarship by JUST.

Currently, she is Full Professor at the College of Engineering, University of Sharjah, UAE, and with the College of Engineering, at the Al-Balqa Applied University, Jordan. She has published about 120 refereed journal and conference papers. She served as Department Chair more than once. She worked at City University of

New York, USA, and Yarmouk University and Jordan University of Science and Technology, Jordan.

Her research interests include: Modeling and Simulation, Geographical Information Systems (GIS), GPS, Remote Sensing, Wireless Navigation Systems, Data Analytics, Transportation Systems, Smart Homes and Cities, Environmental Planning and Engineering, Applied Neural Networks, Passive Solar Design, Quantitative Analysis and Methods, System Recognition and Identification, among others.



Jean François Santucci Jean François Santucci is Full Professor in Computer Science at the University of Corsica since 1996. His main research interests are modeling and simulation of complex systems. He has been author or co-author of more than 150 papers published in international journals or conference proceedings. Furthermore, he has been the advisor or co-advisor of more than 30 Ph.D. students, and he has been involved in the organization of ten international conferences and in the scientific responsibility of five international research projects.

He is conducting newly interdisciplinary research involving modeling and simulation (M&S), machine learning, and Internet of Things (IoT). Since 2014, he is working in new research topics: In the one hand, he is performing researches coupling the Discrete Event System

Specification (DEVS) formalism with IoT, and on the other hand, he investigates how the DEVS formalism could be used in order to improve the reinforcement learning process based on multi-agents features, timing aspects, and hierarchy of abstraction.



Hessam S. Sarjoughian Hessam S. Sarjoughian is an associate professor of computer science and computer engineering in the School of Computing, Informatics, and Decision System Engineering. His research has been funded by NSF, Northrop Grumman, Lockheed Martin, Intel, and Boeing at ASU. Prior to ASU, his research was supported by DARPA, NSF, and the US Airforce. Dr. Sarjoughian's professional engineering experience has been with Honeywell and IBM, spanning nearly five years. He has been integral of editorial board of Sage to interdisciplinary and transdisciplinary national and international projects.

Core research topics are on modeling theories, methodologies, and frameworks that can support the development of composable, heterogenous, multi-scale systems-of-systems. Specialized research areas are poly-formalism modeling, hybrid simulation, agent-based modeling and simulation, collaborative model engineering, and executable software architecture.

Application domain experiences include cyber-physical systems, computation-socio-environmental systems, accelerated network-on-chips, enterprise supply-chain processes, control, and planning, cellular system biology, and service-oriented computing.

Sarjoughian has been leading and developing modeling and simulation frameworks and tools at ASU, including the flagship DEVS-Suite simulator, serving research and development at the science and engineering frontiers.

His research has received four Technical Best-Paper Awards, two Runner-up Technical Best-Paper Awards, and the Best Poster Award at the 2020 Winter Simulation Conference. Sarjoughian has graduated more than sixty masters and doctoral students. He co-founded and co-directs the Arizona Center for Integrative Modeling and Simulation (ACIMS).



Mayank Singh Dr. Mayank Singh is currently working as Sr. Scientist at Consilio Research Lab, Tallinn, Estonia, since January 2020. Prior, he worked as Postdoctoral Fellow, Department of Electrical, Electronic and Computer Engineering, at University of KwaZulu-Natal, Durban, South Africa, and Professor and Head, Department of Computer Science and Engineering at Krishna Engineering College, Ghaziabad, from January 2013 to September 2017. He has 15 + years of extensive experience in IT industry and Academics in India. He completed his Ph.D. in

Computer Science and Engineering from Uttarakhand Technical University in 2011. He obtained his M.E. (2007) in Software Engineering from Thapar University, Patiala, and B.Tech. (2004) in Information Technology from Uttar Pradesh Technical University, Lucknow.

Dr. Singh has organized more than 30 workshops on various technologies and one AICTE-sponsored Faculty development programs (FDPs) on Software Testing, Database Testing, and Reliability.

Dr. Singh has published around 56 research papers in peer-reviewed Transaction, Journals, and Conferences of National and International repute and has supervised 14 M.Tech. and 6 Ph.D. students in the areas of cloud computing, software testing, and ontology and supervising 5 Ph.D. and 2 M.Tech. students. He has published four patents and 1 submitted for grant. He is currently also serving as Life Member of Computer Society of India, Senior Member of IEEE, ACM, and IACSIT.



Claudia Szabo Claudia Szabo is Associate Professor in the School of Computer Science at the University of Adelaide, Australia. Her research interests lie in the area of complex systems and on using simulation to identify and validate their emergent properties. Claudia leads the Complex Systems Program in the Centre for Distributed and Intelligent Technologies and is Associate Editor of the ACM Transactions on Modeling and Computer Simulation. Her email address is claudia.szabo@adelaide.edu.au.



Mamadou Kaba Traoré Mamadou Kaba Traoré is Full Professor at the University of Bordeaux. He got his M.Sc. in mathematical and computational engineering in 1989 and his Ph.D. in computer science in 1992, both at Blaise Pascal University (France). His research interests include the theory of Modeling and Simulation (M&S) and the hybridization of M&S with analytical methods (mamadou-kaba.traore@u-bordeaux.fr). He has published 100+ papers in international journals and conferences and has authored or co-authored a dozen books. Prof. Traoré has been General/Program Chair of several SCS conferences and is currently Associate Editor of the SCS flagship journal (Transactions of the SCS).



Paul Wach Paul Wach is currently working toward a Ph.D. in systems engineering from Virginia Tech and holds a B.S. in biomedical engineering from the Georgia Institute of Technology as well as a M.S. in mechanical engineering from the University of South Carolina. His research interests include theoretical foundations of SE, digital transformation, and artificial intelligence. Paul is a member of the Intelligent Systems Laboratory with the Virginia Tech National Security Institute. He is President and Founder of the Virginia Tech student division of the International Council on Systems Engineering (INCOSE).

He is currently enjoying the full-time student status with part-time association with the Aerospace Corp, where Paul is a senior subject matter expert on the digital transformation. His prior work experience is with the US Department of Energy, two National Laboratories, and the medical industry.



Yan-guang Wang Yan-guang Wang, Ph.D., a senior engineer, graduated from the school of mechanical and vehicle engineering of Hunan University in 2013, majoring in mechanical engineering. From 2003 to 2013, he had obtained bachelor's, master's, and doctor's degrees successively. From 2014 to 2018, he had worked in Computer Technology and Application Institute of Beijing, engaged in inertial navigation product system design and simulation.

Since 2018, he has been working at Beijing Aerospace Intelligent Manufacturing Technology Development Co., Ltd., engaged in technical research in the following fields, such as intelligent manufacturing, digital twin, industrial internet platform, complex system modeling and simulation, cloud simulation. In the past two years, he mainly participated in and completed the application of national projects in smart manufacture area, such as “Network Collaborative Manufacturing and Smart Factory” of the Ministry of Science and Technology, the key project of Sino-German international science and technology innovation cooperation between governments, and the “Industrial Internet Innovation and Development Project” of the Ministry of Industry and Information Technology.



Paul Weirich Paul Weirich is Curators' Distinguished Professor in the Philosophy Department at the University of Missouri. He earned his Ph.D. in Philosophy at UCLA and his B.A. in Philosophy at St. Louis University. His research interests include decision and game theory, logic, and philosophy of science. He uses normative models to create a framework for principles of rational choice.

He is the author of *Equilibrium and Rationality* (Cambridge, 1998), *Decision Space* (Cambridge, 2001), *Realistic Decision Theory* (Oxford, 2004), *Collective Rationality* (Oxford, 2010), *Models of Decision-Making* (Cambridge, 2015), *Rational Responses to Risks* (Oxford, 2020), and *Rational Choice Using Imprecise Probabilities and Utilities* (Cambridge, 2021).

He is a past president of the Central States Philosophical Association, was a visiting fellow at the University of Pittsburgh Center for Philosophy of Science, was the recipient of two grants from the National Science Foundation, and is an associate editor of the *British Journal for the Philosophy of Science*.



Xu Xie Xu Xie received the B.S. degree in Simulation Engineering and the M.S. degree in Control Science and Engineering from National University of Defense Technology, Changsha, China, in 2010 and 2012, respectively. Subsequently, he went to Delft University of Technology, the Netherlands, and got the Ph.D. degree in Modeling and Simulation in 2017. He is currently Associate Professor at National University of Defense Technology. His research interests cover modeling and simulation, discrete event simulation, data assimilation, and transportation.



Chen Yang Chen Yang obtained his Ph.D. in Control Science and Engineering, 2014, and B. Eng. in Automatic and Information Technology, 2008, both from the School of Automation Science and Electrical Engineering and the Honors College (an elite program) of Beihang University (BUAA), Beijing, China. He has worked in the HKU-ZIRI Laboratory for Physical Internet, the University of Hong Kong as a postdoctoral fellow and an associate research officer and in Huawei Technologies, as a senior engineer on R&D tools.

He is currently Associate Professor with the School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing, China. His research

interests include Internet of Things, Industry 4.0, cloud manufacturing, modeling and simulation of complex systems, artificial intelligence, and big data analytics. He is currently serving as Associate Editor of IET Collaborative Intelligent Manufacturing and Editorial Board Member of The Journal of Manufacturing Systems.



Gregory (Greg) Zacharewicz Gregory (Greg) Zacharewicz is Full Professor at IMT—Mines Ales (National Institute of Mines and Telecommunications) in Alès, France. He joined in 2018 the LSR laboratory to develop simulation-driven research works. This laboratory works on the relationship between humans and the complex systems while keeping their roots in the field of Information Science and Technology. He was previously Associate Professor at the University of Bordeaux (2007–2018) where he focused his research for more than 10 years on Enterprise and Social Organization Modelling, Interoperability, and Simulation.

More generally, his research interests include Discrete Event Modeling (e.g., DEVS, G-DEVS), Distributed Simulation, Distributed Synchronization Algorithms, HLA, FEDEP, MDA, Short-lived Ontologies, ERP, BPMN, and Workflow. He recently co-wrote with a worldwide team of authors the prospective chapter “Model-based approaches for interoperability of next generation enterprise information systems: state of the art and future challenges”. In the domain of Healthcare methodologies and technologies, he co-wrote in 2018 with Bernard P. Zeigler, Mamadou K. Traore, and Raphaël Duboz the book “Value-based Learning Healthcare Systems: Integrative modeling and simulation”. He has been the program chair of SpringSim 2016 in Pasadena, the vice general chair of SpringSim 2017 in Virginia Beach, and the general chair of SpringSim 2018 in Baltimore. He is a member of editorial board of Sage [Simulation Journal](#), [JSimE](#), [Science Progress](#), [SNE](#), [Modelling](#), [JKSUCIS](#).



Lin Zhang Lin Zhang is a professor at Beihang University, China. He received the B.S. degree in 1986 from Nankai University, China, and the M.S. and the Ph.D. degrees in 1989 and 1992 from Tsinghua University, China. His research interests include cloud manufacturing and simulation, modeling and simulation for complex systems, model engineering. He served as the president of the Society for Modeling and Simulation International (SCS) and the executive vice president of China Simulation Federation (CSF).

He is currently the president of Asian Simulation Federation (ASIASIM), a fellow of SCS, ASIASIM and CSF, a senior member of IEEE, and a chief scientist of the National High-Tech R&D Program (863) and National Key R&D Program of China. He serves as the

director of Engineering Research Center of Complex Product Advanced Manufacturing Systems, Ministry of Education of China and the editor-in-chief and the associate editors of 6 peer-reviewed international journals. He authored and co-authored more than 300 peer-reviewed journal and conference papers, recognized as Highly Cited Researcher by Clarivate in 2020. He received the National Award for Excellent Science and Technology Books, the Outstanding Individual Award of National High-Tech R&D Program, the National Excellent Scientific and Technological Workers Awards.



Longfei Zhou Longfei Zhou is currently a postdoc associate in the Department of Radiology, Advanced Imaging Laboratories at the Duke University Medical Center. His research interests include smart manufacturing, scheduling, modeling and simulation, machine learning, and computer vision. Dr. Zhou received a B.S. degree in automation from the Harbin Engineering University in 2012 and received a Ph.D. degree in control science and engineering from Beihang University in 2018 where he studied the dynamic scheduling problem in cloud manufacturing. Dr. Zhou joined MIT Computer Science and Artificial Intelligence Laboratory

(CSAIL) in January 2019 as a postdoc to study logistics scheduling, traffic simulation, and manufacturing optimization problems. He joined Duke University in October 2020 as a postdoc associate to do research on machine learning-based medical imaging analysis. Dr. Zhou has published 30+ papers with more than 710 citations. He published a Chinese National Standard as one of the principal drafters and published 2 patents and 2 books. Dr. Zhou's research on modeling and simulation of manufacturing systems was reported by Elsevier's official media. He was granted the Outstanding Contribution Award by the IEEE International Conference on Universal Village in 2020, the Outstanding Doctoral Dissertation Award by the China Simulation Society in 2019, and the Outstanding Graduate by the Beijing Municipal Education Commission in 2018. He serves on editorial boards for 2 journals and served as session chair or program chair for several IEEE, ASME, and CIRP conferences. He is a reviewer of 30+ international journals and 10+ conferences. He served as MIT Postdoc Association Fundraising Chair (2019–2020).

Index

A

Abstraction, 10, 16, 18, 39–42, 63, 109, 127, 195, 196, 208, 229, 233, 234, 238, 290, 354, 356, 365, 367–369, 387, 390, 392, 394, 414–416, 437, 439, 447, 457, 460, 463, 496, 498, 499, 536

Account

causal/mechanical account, 397
deductive/nomological account, 397, 398
pragmatic account, 398
unificationist account, 397, 398, 400

Agent-based model (ABM)

generative ABM, 373, 378
inverse-generative ABM, 378

Aggregate Level Simulation Protocol (ALSP), 30, 156, 160, 477

Analysis methodology, 419

Application Programming Interface (API), 34, 164, 258, 266, 303

Architecture, 18, 34, 55, 60, 63, 64, 107, 111, 125, 126, 136, 152, 153, 160–162, 197, 230, 232, 236, 237, 241, 242, 244, 245, 260–264, 267, 270, 279, 289, 290, 312, 354, 355, 361, 369, 380, 393, 407, 425, 442, 443, 454–458, 460–463, 478, 483, 486, 488, 489, 493, 520, 525, 528, 533, 536

Artificial Intelligence (AI), 100, 107, 110, 112, 113, 132, 159, 169, 189, 193, 235, 271–273, 289–292, 295–297, 361, 402, 404, 472, 477, 478, 481–483, 485–489, 491, 492, 498, 521, 523, 528, 530, 538, 540, 541

Artificial neural networks, 290, 474

Asynchronous programming, 67

B

Bayesian Network, 199, 337, 349, 528

Behavior, 5–7, 9–11, 15, 16, 24, 26–30, 46, 47, 55, 56, 64, 81, 85–88, 91, 93, 94, 96, 99, 107, 114, 115, 133, 136, 144–146, 155, 156, 159, 168, 181, 184–186, 189–191, 196, 197, 199, 217, 233, 236, 239–242, 256–258, 288–290, 292, 293, 295, 298, 300, 301, 303, 304, 313, 330, 348, 354, 355, 359, 365–368, 374, 376–379, 394, 403, 406, 414, 415, 418, 423, 426, 435, 437, 447, 455–461, 465, 473, 475, 480, 481, 483, 492, 499–501

Behavior Modeling, 115, 367, 475

Business Service Model (BSM), 226, 229, 230, 232–234, 236, 237, 239, 242–244

C

Capability Maturity Model Integration (CMMI), 46, 47, 250, 493

Career Program 36 (CP36), 252

Causal

causal explanation, 191, 193, 376
causal modeling, 373
causal models, 81, 373, 375, 379, 380
causal structures, 366
causal variables, 191, 374–376

Causality, 81, 90, 106, 155, 192

Church-Turing, 388

Closure under coupling, 97

Cloud, 30, 37, 49, 60, 101, 110–114, 232, 241, 258, 260–271, 315, 418, 441, 444, 446, 472, 474, 476–479, 485–488, 490, 492, 498, 537, 538

Cloud manufacturing, 450, 527, 540, 541

Combinatorial optimization, 279

Complexity, 10, 12, 46, 51, 99, 109, 130, 132, 182, 223, 267, 270, 276, 279, 290, 312, 354, 356, 359, 369, 415, 438, 440, 453, 458, 462, 477, 481, 484, 496

- Complex problem solving, 82
 Compliance Certification, 153
 Component, 5, 6, 8–11, 14, 16, 27, 43, 46, 49, 54, 56–60, 91, 96, 97, 100, 104, 109, 114, 125, 128, 136, 151, 154, 155, 160, 195, 200, 209, 222, 223, 226, 232, 234–236, 242, 252, 260, 267, 272, 273, 298, 302, 328, 356, 365–368, 376, 386, 387, 397, 403, 415, 419–421, 423, 425, 426, 437, 440, 443, 447, 448, 457–460, 462, 484, 486, 497, 501
 Composability, 26, 54, 57, 60, 386, 437, 440
 Composable models, 154, 366
 Computational Science, 388, 391, 415, 517
 Computer Integrated Manufacturing (CIM), 224
 Conceptualization, 368, 386, 387, 389, 391, 437, 438
 Conceptual model, 41, 45, 63, 101, 103, 185–189, 239, 241, 367, 392, 423–425, 436–439, 447
 Consistent with causality, 90
 Constructive simulation, 2, 23, 24, 122, 126–128, 130–132, 135, 136, 159, 160, 271, 395, 498, 500
 Continuous simulation, 100, 103, 376
 Continuous system, 25, 88–90, 97–99, 198, 314, 415, 481, 523
 Coupled model, 9, 27, 58, 59, 96, 97, 298
 Credibility assessment, 420
 Critical theory, 390
 Cyber Physical System (CPS), 110, 112, 222, 223, 242–245, 260, 261, 263, 407, 464, 474
- D**
 Data assimilation, 448, 449
 Data collection, 389
 Data-driven modelling, 482
 Data model, 18, 35, 37, 264
 Data science, 101, 374, 527
 Dead reckoning, 157, 158
 Decisionmaking, 373, 379, 520
 Deductive-rational method, 389
 Deep uncertainty, 84, 215, 392
 Description, 5, 9, 39, 41–43, 46, 47, 55, 56, 61, 62, 80, 94, 109, 111, 144, 159, 193, 207, 208, 227, 228, 233, 235, 238–240, 242, 256, 257, 262, 268, 316, 369, 374, 376, 377, 385, 396–399, 421, 436, 442, 481, 484, 488, 493, 497
 Design of Experiments (DOE), 300, 369, 419, 420
 Disagreement, 214, 378–381
 Discrete event, 9, 10, 25, 88, 90, 99, 160, 195, 224, 240, 252, 258, 263, 290, 314, 403, 416–418, 449, 453, 458, 460, 461, 482, 484, 497, 516, 517, 523, 526, 535, 539, 540
 Discrete Event System Specification (DEVS)
 Classic DEVS, 97
 Parallel DEVS, 97
 Discrete system, 88–92, 95, 97, 476
 Discretization, 98, 256, 314, 315, 457, 460
 Distributed Interactive Simulation (DIS), 32, 34, 150, 152, 156, 158–162, 359, 477, 531
 Distributed Simulation (DS), 15, 49, 63, 106, 135, 150, 156, 158, 160–162, 240, 252, 260, 302, 356, 358, 359, 386, 425, 478, 540
 Distributed Simulation Engineering and Execution Process (DSEEP), 63, 261, 263, 357–360
 Distribution
 beta distribution, 326, 339–341
 binomial distribution, 316, 319, 323, 324, 327, 328, 333, 341, 343
 Chi-Square distribution, 343–345
 Erlang distribution, 316, 338, 339
 exponential distribution, 316–318, 324, 330, 333, 334, 338
 F distribution, 326, 345–347
 gamma distribution, 331, 335–338, 341, 344
 Gaussian distribution, 316, 319
 geometric distribution, 316, 333–335, 343
 inverse Gamma distribution, 337
 log Normal distribution, 328
 multinomial distribution, 327
 negative binomial distribution, 333, 334, 343
 normal distribution, 319–321, 328, 329, 344, 346
 Pareto distribution, 331–333, 341
 Poisson distribution, 316, 323–325, 328, 335, 347
 Student's t-distribution, 346, 347
 uniform distribution, 326, 327
 Weibull distribution, 316, 329–331, 335
 Diversity, 137, 154, 155, 214, 387, 391, 484
 Domain-Specific Languages (DSL), 42, 98, 263, 270, 367
 Dynamical systems, 263, 312, 448, 355
 Dynamic data driven simulation, 447–449
 Dynamic system model, 449

- E**
- Elapsed duration, 91, 92, 94, 95, 97, 98, 497
 - Empiricism, 389
 - Empirical data, 189, 215, 375, 385, 388, 393, 394
 - Enterprise
 - enterprise interoperability, 230–232, 240, 243, 522
 - enterprise modeling, 222, 224, 237, 244, 519, 526
 - Epistemology, 207, 208, 384–390, 401, 497, 525
 - Error
 - epistemological error, 393
 - error analysis, 178
 - hermeneutical error, 393
 - Empiricism, 389
 - Ethics, 137, 206–215, 384, 394, 402, 465, 497, 523, 525
 - Event
 - single event, 315
 - type of event, 315
 - Evidence, 133, 315, 316, 349, 388, 389, 394, 396, 465, 532
 - Experiment, 2, 8, 24, 33, 78, 79, 87, 89, 111, 112, 186, 188, 296, 300, 301, 324, 328, 334, 341, 354, 388, 392, 394, 395, 401, 402, 419, 420, 422, 447, 448, 460–463, 478, 481, 487, 489, 491, 492, 496
 - Experimental frame, 4, 6–13, 16, 54–57, 59, 85, 127, 154, 182, 186, 194, 195, 199, 300, 392, 418, 419, 455, 456, 461, 496, 498, 500, 501
 - Experimentation, 2, 7, 13, 35, 37, 56, 80, 81, 84, 85, 101, 105–107, 122, 186, 195, 253, 263, 267, 291, 298–302, 359, 393, 395, 407, 421, 447, 463, 464, 496, 523, 531, 534
 - Exploration, 57, 191, 193, 367, 380, 463, 465, 476, 490
 - Exploratory analysis, 84, 130, 191, 193, 214, 378–380, 520
 - External transition, 92–94, 98
- F**
- Factor screening, 420
 - Federate, 34, 63, 161, 162, 232, 235, 238, 272, 274
 - Federation, 42, 63, 110, 122, 135, 159–161, 231, 272, 358, 475–482, 485, 486, 488, 490–492
 - Formalism, 3, 10, 15, 16, 22, 24, 40–42, 56, 88, 90, 95, 97–99, 154–156, 195, 224, 226, 228, 229, 237, 258, 288–290, 314, 386, 395, 402, 417, 418, 455, 458, 460, 461, 499, 516, 517, 536
- Fully specified model, 187**
- Functional Area 57 (FA57), 252**
- G**
- Games**
- digital games, 518
 - serious games, 2, 23, 24, 123, 127, 131, 142, 395, 500
 - simulation games, 142–146, 406, 465
- Grammar, 111, 173**
- Graphical modeling, 229, 238, 367**
- Ground truth, 157**
- Guiding theory, 388**
- H**
- Hierarchical structures, 5, 366
 - High Level Architecture (HLA), 34, 63, 156, 160, 356
- Hybrid**
- hybrid modelling, 107, 532
 - hybrid simulation, 270, 498, 536
 - hybrid situations, 400, 401
- Hybrid modelling, 101–104**
- Hybrid simulation, 101–105**
- Hypothesis**
- acceptance of hypothesis, 396
 - hypothesis generation, 401
 - rejection of hypothesis, 396
- Hysteresis, 173**
- I**
- Inductive-empirical method, 389
- Information and Communication Technology (ICT), 222, 223, 236, 237, 239–242, 472, 482, 528**
- Institute of Electrical and Electronics Engineers (IEEE), 63, 151, 152, 158–160, 213, 356, 358–360, 537, 541**
- Intelligent networks, 481**
- Interaction modeling, 9, 41, 63, 109, 130, 143, 188, 190, 215, 222, 233–235, 239, 265, 279, 290, 362, 379, 404, 416, 444**
- Interactive simulation, 32, 132, 369**
- Internal transition, 92, 94, 95, 97, 98**
- International Organization for Standardization (ISO), 151, 152**
- Interoperability, 3, 10, 26, 63, 64, 150, 153, 154, 158, 195, 222, 230, 231, 234, 237, 238, 240–245, 261, 263, 299, 301, 302, 437, 440, 445, 446, 482, 489, 519, 526, 530, 540**

L

- Learning systems, 304
- Linear time trajectory, 366
- Live, Virtual, and Constructive (LVC), 82, 128, 135, 136, 150, 152, 153, 159, 260, 264, 425
- Live simulation, 2, 23, 24, 122, 128, 133–137, 159, 395, 498, 500

M

- Machine learning, 110, 155, 189, 273, 335, 368, 374, 401, 403, 452, 454, 456, 457, 478, 481, 482, 528, 535, 541
- Markov, 156, 348, 403
- Metamodel, 42, 62, 108, 109, 228, 230, 356, 367, 478
- Mobile applications, 110, 258, 259
- Model
 - model as a service, 48
 - model composition, 26, 49–51, 298, 379
 - model engineering, 22, 40, 42–45, 47, 52, 260, 490, 491, 498, 536, 540
 - model evolution, 42, 47, 499
 - model library, 47, 53, 111, 240, 362, 442, 481, 488
 - model lifecycle, 42–45, 47, 50–52, 420, 498, 499
 - model maturity, 51, 52
 - model reconfiguration, 45, 47, 48
- Model Based Systems Engineering (MBSE), 62, 193, 266, 313, 314, 354
- Model Driven Architecture (MDA), 62, 63, 234, 356, 367
- Model Driven Engineering (MDE), 62–64, 231, 301, 356
- Model Driven Interoperability System Engineering (MDISE), 222, 232–234, 236–239, 242–244
- Model Driven Systems Enterprise Architecture (MDSEA), 63, 222, 223, 225, 229, 230, 233, 234, 236, 239, 244
- Modeling and Simulation (M&S) language, 413
- Modeling and Simulation (M&S) lifecycle, 43–45, 252, 358, 439
- Modeling process, 7, 40, 43, 45–47, 137, 447, 448, 494
- Model transformation
 - model-to-model transformation, 24, 26, 62–64, 154, 228, 239, 243, 299, 301, 356, 499
 - model-to-text transformation, 356

- Multi-disciplinary, 45, 47, 61, 109–111, 114, 137, 276, 277, 291, 392, 405, 464, 465, 474, 478, 482, 487, 488, 521
- Multi-methodology, 103
- Multimodel, 40, 42, 298
- Multi-paradigm, 28, 103, 104, 200, 294
- Multi-technique, 103, 104

N

- Numerical method, 422
- Numerical solution, 337, 415

O

- Ontology
 - methodological ontology, 387, 455
 - referential ontology, 386

P

- Paradigm, 62, 98, 102, 103, 154, 200, 263, 264, 271, 296, 318, 341, 356, 374, 381, 386, 392, 398, 405, 421, 424, 426, 447, 449, 465, 485, 498
- Parallel evolutionary algorithms, 274, 278, 279
- Parallel transfer evolution, 278, 279
- Phenomenon, 10, 81, 209, 256, 317, 374, 388, 395, 396, 399, 402, 463, 498, 499
- Philosophy, 23, 29, 80, 104, 206–208, 210, 384, 385, 387, 390, 397, 399, 400, 438, 497, 523, 525, 539
- Physical science, 355, 374, 375, 385, 465
- Planning
 - strategic planning, 379, 419, 520
 - tactical planning, 419
- Positivism, 385, 390
- Postdiction, 191, 193, 376
- Prediction, 24, 30, 36, 37, 64, 79, 87, 111, 114, 115, 186, 189, 191–193, 329, 374–378, 380, 381, 391, 394, 399, 414, 419, 420, 446, 454, 474, 492
- Predictive models, 11, 25, 81, 87, 105, 189, 192, 375, 376, 396, 446, 499, 528
- Prescriptive analytics, 105
- Probability
 - axioms of probability, 316
 - conditional probability, 316, 334
 - evidential probability, 315, 316
 - objective probability, 316, 349
 - physical probability, 315, 316
 - prior probability, 199, 349
 - subjective probability, 316
- Professional codes, 210, 212, 213, 497
- Professional Education, 251, 531

- Proposition, 237, 242–244, 385, 388, 393, 394, 397
- Protocol Data Unit (PDU), 32, 158, 159, 161
- Q**
- Qualitative modeling, 377
- Quantization, 315
- Quantized state systems, 315
- R**
- Ranking & Selection, 420
- Rationalism, 389
- Real-time
- real-time scheduling, 450
 - real-time simulation, 49, 451, 527
 - real-time simulation experimentation, 104, 106
- Referent, 187, 189, 386, 439, 440
- Regress, 388, 394
- Relativity, 14, 390
- Reproducibility, 377, 396
- Response surface methodology, 419, 420, 424
- Reuse, 46–49, 51, 53–55, 63, 153, 154, 156, 194, 302, 437, 439, 440, 489, 522
- Run-Time Infrastructure (RTI), 161, 162, 442, 445
- S**
- Scientific method, 78, 389–392, 394
- Scientific test, 396
- Semantics, 10, 18, 42, 53, 156, 263, 417, 489
- Semi-Automated Forces (SAF), 159, 160
- Sequential Bifurcation, 420
- Simuland, 4, 15, 16, 182, 185, 191, 402, 500, 501
- Simulation animation, 510
- Simulation Interoperability Standards Organization (SISO), 152, 158–160, 261, 262, 270, 517, 531
- Simulation systems engineering, 260, 354, 356
- Simulator Networking (SIMNET), 156
- Social science, 132, 133, 189, 191, 312, 331, 355, 373–375, 378, 390, 398, 402, 407, 421, 438, 463, 465, 475, 480, 517, 520
- Soft system thinking, 377
- Specification
- De Facto standards, 150, 496
 - De Jure standards, 150, 497
 - differential equation specification, 56
 - discrete-time specification, 56
 - discrete-event specification, 314
 - proprietary standards, 150, 151, 499
 - open standards, 151, 160
- Standards Developing Organizations (SDO), 151, 152
- Standards Setting Organizations (SSO), 151
- Stochastic differential equations, 151, 160
- Superdense time trajectory, 155
- Supply chain, 146, 162, 200, 201, 222, 223, 238–242, 427
- Synchronisation protocols, 106
- Synthetic Environment Data Representation and Interchange Specification (SEDRIS), 152, 153, 503
- System dynamics, 25, 40, 82, 101, 104, 184, 190, 312, 366, 377, 415, 498
- System Entity Structure (SES), 3, 16, 18, 57–60, 387, 460–462, 489, 524, 535
- System lifecycle processes, 44
- System of Systems (SoS) Engineering, 260, 355
- T**
- Technology Independent Model (TIM), 226, 230, 232, 234, 236, 239, 240, 242, 244, 245
- Technology Specific Model (TSM), 232–234, 236, 241, 242, 244
- Test and Training Enabling Architecture (TENA), 150, 152, 156, 159, 162, 359
- Testing, 24, 38, 60, 63, 101, 124, 125, 184, 189, 190, 253, 257, 346, 359, 368, 379, 396, 400, 401, 407, 426, 443, 454, 531, 537
- Textual languages, 366
- Time
- physical time, 10, 14, 15, 88, 115, 154, 235, 238, 244, 257, 272, 279, 295, 314, 361, 442, 490, 498, 499, 501
 - real-time, 14, 15, 29, 34, 37, 49, 65, 104–107, 111, 115, 125, 127, 136, 158, 198, 265, 272, 273, 361, 363, 364, 441, 442, 446–449, 451, 453, 456, 474, 480, 482, 484, 486–490, 492, 493, 527, 532
 - simulated time, 15, 115, 134, 136, 188, 314
 - virtual time, 6, 111, 115, 136, 142, 361, 363, 480, 489, 490
 - wall-clock time, 14, 106, 499, 501
- Time advance, 92, 95, 196, 289
- Total state, 91–93, 97, 497
- U**
- Uncertainty, 45, 46, 80, 81, 83, 84, 190, 195, 207–209, 214, 288, 367, 378, 379, 392, 420, 476, 482

V

Validation, 16, 62, 98, 105, 153, 159, 160,
182–185, 189–191, 193, 200, 208, 209,
244, 261, 359, 368, 396, 419, 425, 460,
489, 491, 492

Variance-reduction techniques, 419, 420

Verification, 16, 42, 43, 54, 62, 98, 105, 112,
153, 159, 160, 182–185, 193, 195–197,
200, 208, 209, 244, 261, 365, 425, 491,
492

Virtualization, 115, 265–268, 272, 276, 441,
487, 490

Virtual simulation, 2, 23, 24, 122, 123,
125–127, 135, 159, 267, 395, 498, 500

Visual analytics, 366, 368, 369, 528

Visual interactive simulation, 368

Visualization

component visualization, 112, 366, 486,
487

dynamic visualization, 111, 367, 368, 391,
486

multi-dimensional visualization, 365

Visualization modeling frameworks, 442

Visual languages, 366, 367

Visual modeling, 365–369, 481, 525

W

Web server, 324, 347

Web services, 60, 234, 258, 259

Worldview, 388, 390, 394, 405, 438